Universidad Carlos iii de Madrid

Escuela Politécnica Superior

Ingeniería Industrial

PROYECTO FIN DE CARRERA

TELEOPERACIÓN DE UN ROBOT MINDSTORM MEDIANTE TÉCNICAS DE VISIÓN ARTIFICIAL

Autor: Eduardo Méndez Rodríguez
Tutores: Jorge García Bueno
Piotr Pawel Jurewicz Slupska

Julio de 2011

A mi familia y mis amigos, sin su apoyo no podría haberlo logrado.

Agradecimientos

A mis padres, por estar siempre a mi lado y no dejar de creer en mí en ningún momento. También debo darles las gracias por animarme y empujarme en los momentos oportunos dándome el impulso necesario para convertir mis sueños en realidad.

A mi hermano, por estar ahí siempre que ha hecho falta y ayudarme en todo lo que ha podido.

A mis compañeros de piso, que siempre han hecho más llevaderos los malos días.

A mis profesores y compañeros de estudio, sin los que llegar a este momento hubiese sido mucho más costoso.

Eduardo Méndez Rodríguez.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

CAPÍTULO 1

INTRODUCCIÓN

En esta introducción se exponen los dos conceptos básicos y necesarios para el completo entendimiento de la finalidad del proyecto. Estos conceptos son los de teleoperación y visión artificial.

1.1. Teleoperación

La teleoperación es una disciplina que esta ligada a otras como la automática, informática, electrónica, mecánica, etc. La teleoperación permite gobernar un dispositivo esclavo (controlar tanto su movimiento como la fuerza ejercida) ubicado en una zona remota a través de un dispositivo maestro localizado en el punto de trabajo del operador. Así, el movimiento del esclavo debe ser predecible a partir de los movimientos que el operador realice.

Las investigaciones referentes a la telemanipulación de objetos nacieron en los laboratorios de la industria nuclear, debido al alto riesgo que presenta estar en contacto directo con los elementos radioactivos. De esta forma en 1947 comenzaron las primeras investigaciones, lideradas por Raymond Goertz del

Argonne National Laboratory en Estados Unidos, encaminadas al desarrollo de algún tipo de manipulador de fácil manejo a distancia mediante el uso por parte del operador de otro manipulador equivalente. El primer fruto se obtuvo en 1948 con el desarrollo del primer manipulador teleoperado mecánico, denominado M1, antecesor de toda la familia de sistemas maestro-esclavo de telemanipulación existentes actualmente.

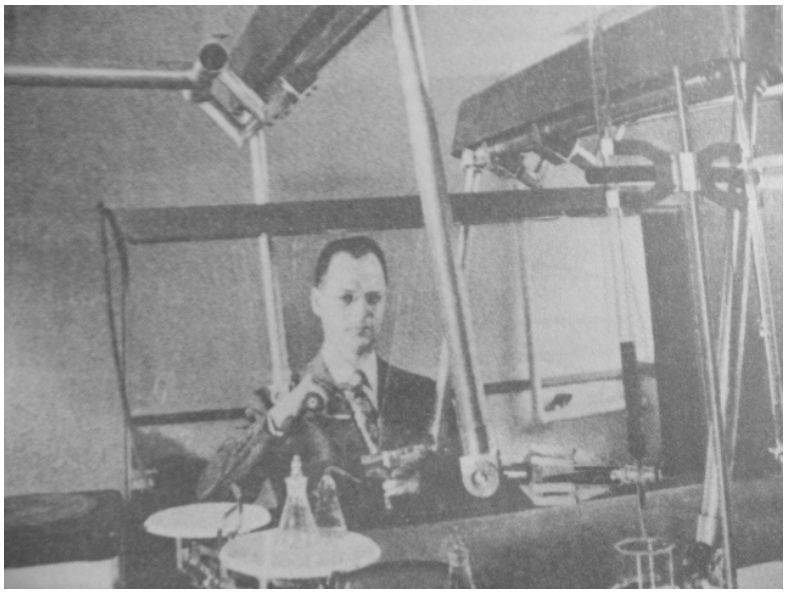

Figura 1.1: Primer manipulador teleoperado mecánico, M1.

El mecanismo de este sistema permitía que la pinza situada en el extremo del manipulador esclavo reprodujera de forma fiel los movimientos hechos por la mano del operador al extremo del manipulador maestro. Ambos manipuladores eran prácticamente iguales, y los movimientos entre ambos se reproducían eje a eje, de tal manera que el extremo de ambos describiese la misma trayectoria.

A principios de los años 50 se comenzaron los desarrollos encaminados a motorizar ambos manipuladores, maestro y esclavo, de una forma adecuada. Fue en 1954 cuando Raymond Goertz presentó el primer manipulador maestro-esclavo con accionamiento eléctrico y servocontrol en ambos manipuladores llamado E1.

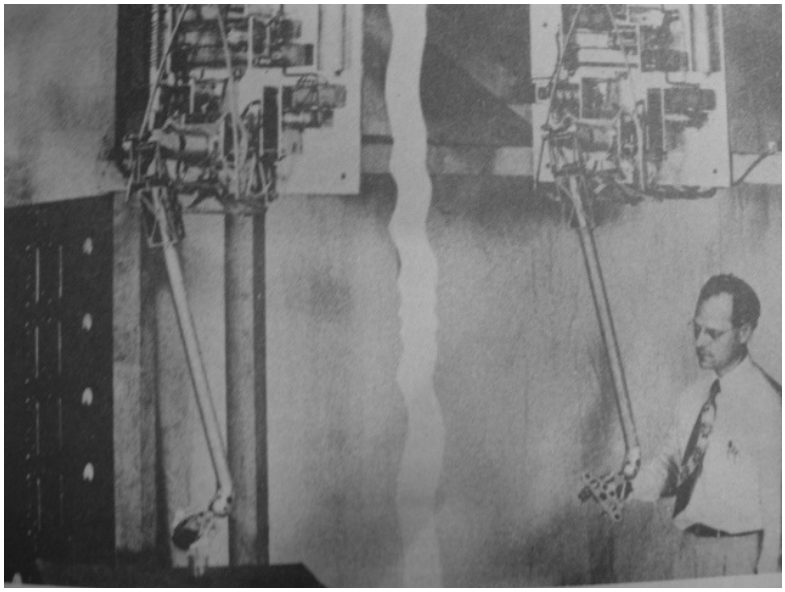

Figura 1.2: Primer manipulador teleoperado eléctrico, E1.

A finales de los años sesenta y principios de los setenta, la tecnología de la teleoperación alcanzó su mayoría de edad con su utilización en aplicaciones espaciales. Aparecieron nuevos retos y problemas, siendo de especial relevancia la existencia de retrasos temporales en la comunicación entre la zona local y la zona remota.

En paralelo con la evolución histórica de las técnicas de teleoperación ha habido una evolución tecnológica motivada por los desarrollos de control, la informática y la robótica. Ha habido, por una parte una evolución en los sistemas de comunicación, pasando de los sistemas mecánicos a los eléctricos, fibra óptica, radio e Internet, medio que suprime prácticamente las limitaciones de distancia.

1.2. Visión Artificial

La visión artificial o visión por computador es una disciplina compleja que involucra otras ciencias e incluye estudios de física, matemáticas, ingeniería electrónica... El continuo desarrollo de algoritmos, funciones y aplicaciones hace que sea una disciplina en continua evolución. La visión artificial tiene

por objetivo modelar matemáticamente los procesos de percepción visual de los seres vivos y generar programas que permitan simular estas capacidades visuales por ordenador.

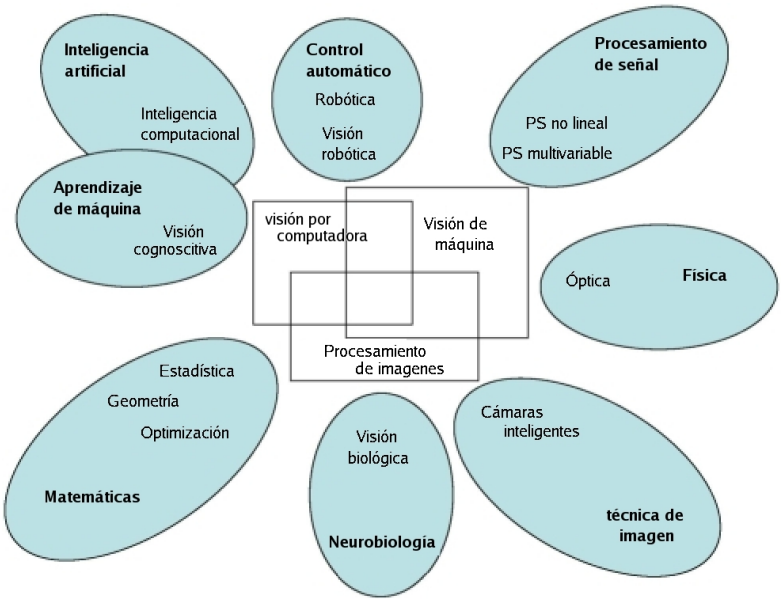

Figura 1.3: Relación de la visión por computador y otras áreas afines.

No es hasta la década de los 50 cuando empiezan a aparecer los primeros trabajos relacionados con la visión artificial. Al principio se piensa que es una tarea sencilla y alcanzable en pocos años, esto se debe a los importantes trabajos realizados por Roberts (1963) y Wichman (1967). El primero demuestra la posibilidad de procesar una imagen digitalizada para obtener una descripción matemática de los objetos que aparecían y el segundo presenta por primera vez una cámara de televisión conectada a un ordenador. Esta ilusión del principio propiciada por los avances comentados, los avances en los ordenadores y por la creencia de que si para los humanos ver es una tarea fácil para un ordenador también lo será, pronto se empieza a oscurecer debido a los limitados avances conseguidos y a que estos sólo se pueden aplicar a un número reducido de imágenes, además se observa como la visión que es una tarea que parece relativamente trivial para los humanos, es muy compleja de llevar a cabo por ordenadores. Es por ello que los años setenta no presentan avances significativos ya que se produce un continuo abandono de las investigaciones.

En la década de los ochenta vuelven a aparecer las investigaciones relacio-

nadas con la visión por computador en este caso encaminadas a la extracción de características. Así se tiene la detección de texturas Haralik (1979) y la obtención de la forma a través de ellas Witkin (1981). Además en esta década se publican también numerosos artículos entre los que destacan: visión estéreo (Mayhew y Frisby), detección de movimiento (Horn), interpretación de formas (Steven) y detectores de esquina (Kitechen y Rosendfekd). A pesar de la importancia de las investigaciones y artículos recién comentados, el trabajo más importante de la década es el libro de David Marr «Vision: a computacional investigation into the human representation and processing of visual information», donde se abordaba por primera vez una metodología completa del análisis de imágenes a través de ordenador.

El proceso de visión por ordenador puede subdividirse en seis áreas principales:

- 1. Sensado. Es el proceso que nos lleva a la obtención de una imagen visual.
- 2. Preprocesamiento. Trata de las técnicas de reducción de ruido y enriquecimiento de detalles en la imagen.
- 3. Segmentación. Es el proceso que divide una imagen en objetos de interés.
- 4. Descripción. Trata con el cómputo de características útiles para diferenciar un tipo de objeto de otro.
- 5. Reconocimiento. Es el proceso que identifica esos objetos.
- 6. Interpretación. Asigna un significado a un conjunto de objetos reconocidos.

Los objetivos más típicos que se pretenden satisfacer mediante la visión artificial son:

- La detección, segmentación, localización y reconocimiento de ciertos objetos en imágenes.
- La evaluación de los resultados.
- Registro de diferentes imágenes de una misma escena u objeto.

- Seguimiento de un objeto en una secuencia de imágenes.
- Mapeo de una escena para generar un modelo tridimensional de la escena, tal modelo podría ser usado por un robot para navegar por la escena.
- Estimación de las posturas tridimensionales de humanos.
- Búsqueda de imágenes digitales por su contenido.

CAPÍTULO 2

ESTADO DEL ARTE

En este capítulo se muestra el estado del arte de las tres partes tratadas en este proyecto. Estas partes son la teleoperación (principalmente con comunicaciones mediante bluetooth), la visión artificial para elementos móviles en tiempo real y proyectos realizados con el Mindstorm NXT. Hay que destacar que en los dos primeros campos hay innumerables aplicaciones debido a que ambos campos siguen hoy en día bajo invetigación en busca de lograr mejoras tecnológicas, también los proyectos con el Mindstorm están creciendo exponencialmente, mayormente debido a su uso en el campo de la enseñanza en las universidades.

2.1. Teleoperación

En la actualidad la teleoperación es ampliamente utilizada y se sigue investigando en múltiples campos. Éstos van desde conseguir realizar acciones peligrosas mediante robots que hoy en día se siguen realizando por personas hasta avances en el mundo del entretenimiento. A continuación se muestran varios proyectos interesantes e innovadores donde se puede apreciar la mag-

nitud de la teleoperación en la actualidad.

2.1.1. Helicóptero para monitorización aérea en el sistema multi-UAV COMETS

COMETS es un proyecto del programa IST (Sociedad de Información Tecnológica, del inglés Information Society Technologies) de la Comisión Europea. Su principal objetivo es la coordinación y control en tiempo real de una flota heterogénea de vehículos aéreos no tripulados (UAV). Para la realización de este proyecto se diseñó un helicóptero con las especificaciones y tamaño requeridas para llevar a cabo sus tareas.

Figura 2.1: Helicóptero diseñado para el proyecto COMETS.

La comunicación con el helicóptero se consigue a través de un enlace inalámbrico mediante los protocolos TCP/UDP/IP. Puesto que se proporcionan vídeos el ancho de banda es un factor crítico por lo cual la distancia máxima de correcta operación se verá limitada por factores como la meteorología o el relieve del terreno en la zona de vuelo.

Los primeros resultados experimentales generales se realizaron en mayo de 2003 de manera positiva. El sistema de teleoperación cumplió las expectativas en cuanta a la integración en el sistema COMETS y cumplimiento de misiones.

2.1.2. Telemedicina

Un campo en el que la introducción de la teleoperación ha sufrido una evolución considerable es el de la medicina. Ese avance se puede apreciar en las llamadas cirugías laparoscópicas en las cuales mediante pequeñas incisiones de 5 milímetros difícilmente perceptibles se realizan operaciones sin necesidad de tener que abrir al paciente para ver lo que sucede en la zona de operación. Otra de las ventajas es que se consigue más precisión pues se puede controlar cada movimiento regulando la sensibilidad del sistema y se evitan gran parte de los problemas de cansancio del cirujano en largas operaciones.

Figura 2.2: Robot de quirófano Da Vinci para realizar operaciones laparoscópicas.

Cabe destacar que para este tipo de operaciones todo el instrumental y equipo quirúrgico con sus controles debe estar totalmente integrado en la sala de operaciones de una forma ergonómica. Una vez el quirófano está adaptado, la teleoperación se puede realizar dentro del mismo quirófano o mediante el uso de tecnologías de redes inalámbricas que hace posible que la teleoperación se realice desde otro país o incluso otro continente.

Se pueden distinguir dos tipos de teleoperación:

- Telecirugía asistida: mediante una comunicación permanente por televisión entre un experto y el cirujano que está realizando la operación,

aconsejar y dirigir los gestos que han de realizar en cada momento para el buen éxito de la operación.

- Telerobótica: permite a un cirujano situado en otro hospital, ciudad, país o continente dar las órdenes precisas a un robot para que realice la operación que se requiere.

2.1.3. ShakerRacer

Este es un ejemplo de aplicación realizada por aficionados dentro del mundo del ocio en la teleoperación. Este proyecto consiste en el manejo de un vehículo por control remoto mediante un teléfono móvil gracias a una comunicación bluetooth.

El control del vehículo se consigue con un programa que utiliza el acelerómetro del teléfono móvil y en función del resultado se ejecuta una orden. En la siguiente figura se puede ver como en la pantalla del teléfono aparece un punto rojo que es la posición gráfica del acelerómetro. Inicialmente esta en reposo y al girar el teléfono móvil a la izquierda el acelerómetro se desplaza a la derecha, moviéndose las ruedas del vehículo en el sentido que se ha movido al móvil.

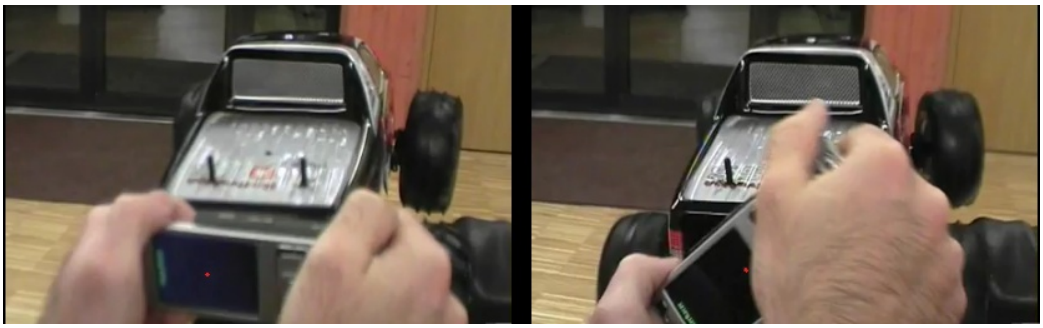

Figura 2.3: Ejemplo de movimiento del vehículo.

2.2. Visión artificial por computador

Las aplicaciones de la visión artificial en la actualidad son muy variadas e interesantes, a continuación se muestran algunas de ellas:

- Industria automotriz: medición de las dimensiones de cojinetes de frenos, calibración de ensamblado robótico de sensores de frenos 'anti-lock'.

- Industria de dispositivos médicos: inspección de catéteres en el corazón, lectura de códigos en marcapasos.

- Industrias financieras: inspección detallada de tarjetas financieras.

- Retroalimentación visual para robots.

- Comunicación visual hombre-máquina.

- Empresas de seguridad: videovigilancia.

- Control de tráfico.

Puesto que para la realización de este proyecto hay que detectar la posición de la mano en tiempo real se exponen a continuación proyectos que trabajan con reconocimiento de gestos.

2.2.1. Reconocimiento de gestos del lenguaje de sordos

Uno de los proyectos de visión artificial que se ha realizado, en la Universidad Politécnica de Madrid en el año 2000, es el reconocimiento de los patrones de las manos del lenguaje de sordos. Para la resolución de este proyecto se realiza una red neuronal. Una vez captada la imagen ésta se acondiciona mediante una serie de operaciones morfológicas (almacenamiento de píxeles de frontera, erosión, dilatación, transformación de distancia y esqueletización). Tras el acondicionamiento se procede al procesamiento de la imagen con la cual se consiguen una serie de características (área de la mano, número de agujeros, perímetro de la mano, curvatura, número de dedos, circularidad, ángulo de inclinación y relación de aspecto) que se tendrán en cuenta a la hora de clasificar un gesto.

Figura 2.4: Ejemplo de interpretación de signo de lenguaje de sordos mediante visión artificial.

En este proyecto se decidió usar un guante blanco y un fondo oscuro con la finalidad de que la implementación y el entrenamiento de la red neuronal se pueda realizar de manera mas sencilla, evitandose sombras en el fondo y pudiendo eliminar el antebrazo en el análisis debido a la diferencia de tono.

Hay que resaltar que el proyecto mencionado es sólo orientativo para la clasificación de los gestos ya que en la realidad hay que tener en cuenta otros factores además de las manos como pueden ser la posición de la mano relativa al cuerpo, el movimiento de la mano o componentes no manuales como movimiento de los labios o la expresión facial.

2.2.2. HandVu

HandVu es un software incluido en OpenCV (biblioteca libre de visión artificial originalmente desarrollada por Intel) en el cual se implementa un interfaz de visión basada en los gestos de la mano. Esta interfaz detecta la mano en una postura estándar y entonces la sigue reconociendo gestos clave, y todo esto en tiempo real sin necesidad de calibración de la cámara. Dentro del paquete se encuentra la librería principal de HandVU y numerosas aplicaciones que muestran las funcionalidades del programa.

En la siguiente figura se puede apreciar como mediante el movimiento de

la mano se mueven las bolitas, siempre siendo reconocida la mano (enmarcada).

Figura 2.5: Ejemplo de aplicación de HandVU.

2.3. MindStorm NXT

El LEGO MindStorm NXT en la actualidad cada vez se utiliza más en centros educativos relacionados con la ingeniería de control y automática, ya sean cursos que van desde unas pocas horas al semestre hasta proyectos de final de carrera. Una muestra de ellos son *RWTH Aachen University* (Alemania), *Otto-von-Guericke University Magdeburg* (Alemania), *University of Cambridge* (Gran Bretaña), *New Jersey Institute of Technology* (Estados Unidos), *University of Illinois at Chicago* (Estados Unidos), *Radboud University Nijmegen* (Holanda), *Lund University* (Suecia) o *University of Ottawa* (Canadá). Esto se debe a la facilidad de comprender su funcionamiento y la simplicidad de programarlo con los fines deseados.

Hay innumerables proyectos realizados en las diversas universidades que trabajan en la actualidad con el Mindstorm NXT, un ejemplo de los más significativos se muestran más abajo. También cabe destacar que hay innumerables aplicaciones realizadas anónimamente y publicados en foros acerca del tema.

2.3.1. Robot controlado por gestos

Este trabajo fue realizado como proyecto de final de carrera en la Universidad de Illinois en Estados Unidos y consiste en controlar un vehículo móvil mediante la realización de gestos con el brazo. Para lograr detectar el movimiento del brazo se hará uso del Wiimote (mando de la consola Wii de

Nintendo) el cual consta de tres acelerómetros, uno en cada eje, de manera que se pueden medir las aceleraciones horizontal, vertical y en profundidad pudiendo calcular una aproximación del movimiento o gesto realizado.

Figura 2.6: Dispositivo utilizado para captar el movimiento (Wiimote).

La idea de funcionamiento es que el bloque NXT mediante la información obtenida del Wiimote proporcione a los motores del vehículo las acciones necesarias para que se mueva de la misma forma que el gesto que ha captado. La comunicación se realizará entre el Wiimote y el procesador (un ordenador) mediante bluetooth y éste manda la información al bloque NXT mediante el puerto USB. Sabiendo esto el esquema del modelo se puede expresar de la forma mostrada a continuación.

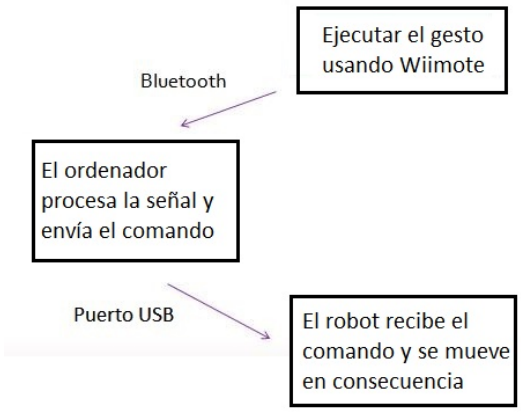

Figura 2.7: Proceso de comunicaciones del robot controlado por gestos.

Para la lectura y el análisis de las señales del Wiimote se utiliza el Matlab, y más concretamente se hace uso de la toolbox WiiLab (realizada por la Universidad de Notre Dame).

2.3.2. Robot embarcación

Este proyecto fue realizado por el Doctor Omar Sánchez, actualmente profesor de la Universidad de Huelva, y consiste en crear un robot embarcación que sea autónomo. Esto se consigue usando varios dispositivos a la vez, los principales son el robot Mindstorm NXT, un GPS, ordenador con bluetooth y conexión a internet, sonar y una webcam.

Figura 2.8: Embarcación utilizada y disposición del bloque NXT.

Dentro de la embarcación se utilizaran dos motores mediante el bloque NXT, uno de ellos manejará las hélices para el avance de la embarcación y el otro motor hará moverse el timón para poder realizar los giros deseados. En la parte trasera de la embarcación se pueden apreciar las salidas de los motores.

Figura 2.9: Uso de los motores en la embarcación

La idea es que la embarcación sea autónoma y se mueva mediante el control de un procesador situado en el ordenador, en el que se recogerán

todos los datos captados por los distintos dispositivos: GPS, webcam y sonar. El ordenador estará conectado al Mindstorm mediante conexión bluetooth y a su vez enviará información de los sensores por internet para en caso de necesidad poder controlarlo remotamente.

2.3.3. Robot solucionador de sudokus

Este dispositivo mediante una estructura específica que permita pasar las páginas del libro donde se encuentran los sudokus, una webcam y un ordenador con un procesador de imágenes y de datos. La estructura se muestra en la siguiente figura.

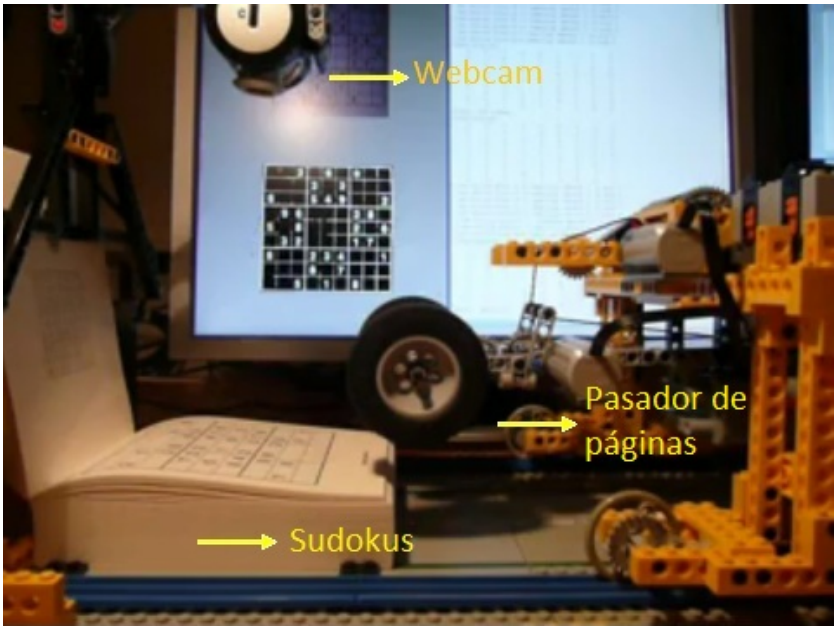

Figura 2.10: Ubicación de las partes del robot solucionador de sudokus.

El proceso es que el Mindstorm pasa la página y mediante la webcam y un procesador de imágenes se guarda en el ordenador, donde se interpretará la imagen captada por la webcam y se solucionará el sudoku. Una vez esto pase se envía una señal al Mindstorm para que pase la página y se continúe con el siguiente sudoku.

CAPÍTULO 3

ARQUITECTURA HARDWARE Y SOFTWARE

A continuación se expone toda la información de las herramientas y dispositivos utilizados para la realización de la teleoperación dentro de este proyecto.

3.1. Sistema Operativo

En el presente proyecto se ha utilizado el sistema operativo Windows 7, del cual se tiene habitualmente la licencia al adquirir un ordenador. Este entorno actual tiene unas características de facilidad de uso y de estabilidad necesarios para la ejecución de este proyecto.

Cabe destacar que se pueden usar otros sistemas operativos para la consecución de este proyecto como pueden ser Linux o Mac OS puesto que todos los programas y dispositivos que se explican a continuación tienen una versión compatible con estos sistemas operativos.

3.2. Matlab R2010b

Matlab es un software matemático que ofrece un entorno de desarrollo integrado con un lenguaje de programación propio (lenguaje M) el cual está disponible para los sistemas operativos expuestos en el apartado anterior. En concreto se utiliza la versión R2010b, debido a la necesidad de trabajar con la versión del programa de 32-bits (esta versión es la primera que funciona con 32-bits sobre un sistema operativo de 64-bits) puesto que el microprocesador del Mindstorm no es capaz de conectar aún con un programa de 64-bits.

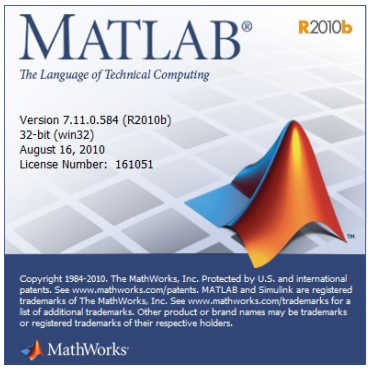

Figura 3.1: Matlab R2010b.

Dentro de esta aplicación se encuentran unos paquetes de herramientas llamados *toolboxes* de los cuales se usan tres de ellos:

- Image Acquisition: Esta *toolbox* contiene herramientas para la adquisición de imágenes y funciones que extienden la capacidad de Matlab de computación numéricas. Dentro de esta *toolbox* se encuentran funciones como por ejemplo, adquisición de imágenes a través de dispositivos externos (por ejemplo una cámara web), vista previa de un video en tiempo real, configuración de funciones que se ejecuten automáticamente cuando ciertos eventos ocurran, llevar información de la imagen al entorno de trabajo de Matlab...
- Image Processing: Al igual que la herramienta anterior también ésta extiende las capacidades para trabajar en un ambiente de computación numérica. En este caso las funciones disponibles, entre otras, son las

transformaciones espaciales de la imagen, operaciones morfológicas, filtros lineales y filtros diseñados, transformaciones, análisis de imágenes y su mejora, registro de imágenes, operaciones en regiones de interés...

- RWTH-Mindstorm NXT: esta *toolbox* fue realizada por la Universidad de Aachen con fines académicos pero al ver los buenos resultados optaron por licenciarla en GNU, de manera que es un código abierto y gratuito. Actualmente la compañía Mathworks (creadora de Matlab) la ha validado y se puede bajar directamente de su página web oficial. Dentro de este paquete de herramientas se proveen de funciones para la comunicaciones con el robot LEGO® Mindstorm NXT tanto vía USB como vía bluetooth. Esta toolbox incluye rutinas para soportar la interacción entre el robot y MATLAB. Se pueden encontrar funciones para abrir y cerrar conexiones, enviar y recibir datos entre el robot y MATLAB, controlar los motores del robot, leer los sensores del robot...

3.3. LEGO Mindstorm NXT

Lego Mindstorms es un juego de robótica fabricado por la empresa Lego, el cual posee elementos básicos de las teorías robóticas, como la unión de piezas y la programación de acciones, en forma interactiva. Este robot fue comercializado por primera vez en septiembre de 1998. A parte de como juego también se vende como herramienta educacional, lo que originalmente se pensó en una sociedad entre Lego y el MIT finalmente acabo con Lego financiando la investigación y pudiendo usar las aplicaciones creadas sin pagar costes al MIT.

Lego Mindstorms puede ser usado para construir un modelo de sistema integrado con partes electromecánicas controladas por computador mediante un software de programación basado en una interfaz gráfica para usuarios. Prácticamente todo puede ser representado con las piezas tal como en la vida real, como un elevador o robots industriales.

Al adquirir el paquete completo del Lego Mindstorm NXT se pueden diferencias dos partes: las partes físicas (actuadores, sensores, bloque NXT y las piezas para los montajes) y el software.

Figura 3.2: Paquete LEGO Mindstorm NXT 2.0.

3.3.1. Partes físicas

A continuación se muestran las partes físicas de Lego Mindstorm NXT desglosadas.

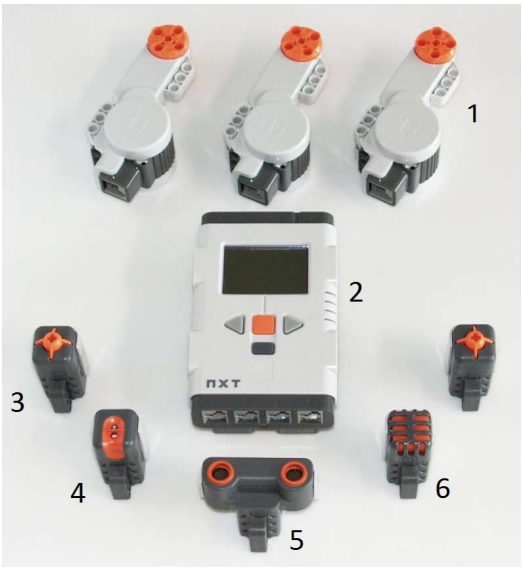

Figura 3.3: LEGO Mindstorm NXT.

- 1. Tres servomotores para proporcionar movimiento a las estructuras montadas.
- 2. Bloque NXT en el cual se encuentra el microprocesador y las salidas y entradas de los sensores y actuadores.

- 3. Dos sensores presión. Estos sensores, del tipo todo o nada, sirven para detectar contacto con objetos.

- 4. Sensor de color. Puede detectar 6 colores diferentes, además también puede ser usada como una linterna de colores.

- 5. Ultrasonido. Provee de información de la distancia del objeto más cercano a su ángulo de visión. Tiene un rango entre 0 y 255 centímetros.

- 6. Sensor de sonido. Sirve de micro para grabar sonidos en decibelios (dB). Se puede ajustar entre dos modos: decibelios (micro) y decibelios ajustables (la sensibilidad del sensor se va ajustando como lo haría la del oído humano).

Hay que destacar que sólo se han comentado las partes físicas activas, aparte de lo mostrado están las piezas con las que se montan las estructuras de los modelos creados.

3.3.2. Software

Junto a las piezas ya mencionadas en el apartado anterior se adquiere un software de fácil uso creado específicamente para el control del robot Mindstorm NXT. Desde este software se puede conectar con el boque NXT y crear programas de diversa dificultad en un cómodo entorno gráfico de bloques.

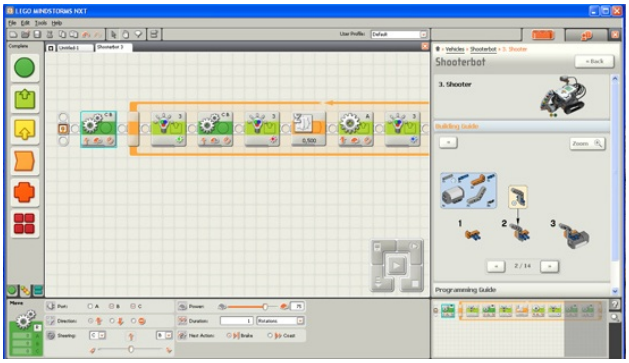

Figura 3.4: Software de Lego Mindstorm NXT.

3.4. Dispositivos

3.4.1. Webcam

El éxito o no de un algoritmo de visión por ordenador depende en gran medida de la calidad de la imagen sobre la que se trabaja, llegando incluso a ser más importante para el éxito de la aplicación que el propio algoritmo. Es por ello que en este proyecto se ha trabajado con una cámara de video Logitech Quickcam E1000, puesto que no hay necesidad de resoluciones muy grandes debido al tiempo que conllevaría su proceso mediante MATLAB. Las características que más interesan de este dispositivo son:

- Sensor de 1,3 megapíxeles reales (1280 x 1024 píxeles).
- Captura de vídeo: hasta 1280 x 1024 píxeles.
- Vídeo de hasta 30 cuadros por segundo.
- Certificación USB 2.0 de alta velocidad.
- Clip universal que se acopla a la mayoría de monitores y pantallas de portátiles.
- Enfoque manual.

Cabe destacar que aunque el enfoque no sea automático como la cámara va a estar en un punto fijo es muy fácil ajustarla manualmente y una vez hecho mientras no se mueva el punto de trabajo no habrá necesidad de reconfigurarla.

Figura 3.5: Logitech Quickcam E1000.

3.4.2. Bluetooth

Se denomina bluetooth al protocolo de comunicaciones inalámbrico diseñado especialmente para dispositivos de bajo consumo, con una cobertura baja y basados en transceptores (dispositivos con el emisor y receptor en la misma caja, compartiendo partes del circuito electrónico) de bajo coste. Gracias a este protocolo, los dispositivos que lo implementan pueden comunicarse entre ellos cuando se encuentran dentro de su alcance. Las comunicaciones se realizan por radiofrecuencia en la banda ISM (Industrial, Scientific and Medical. Son bandas reservadas internacionalmente para uso no comercial) de los 2,4Ghz de forma que los dispositivos no tienen que estar alineados y pueden incluso estar en habitaciones separadas si la potencia de transmisión lo permite.

Los dispositivos con Bluetooth también pueden clasificarse según su ancho de banda. Por las especificaciones del LEGO Mindstorm NXT el dispositivo de bluetooth tiene que tener al menos la versión 2.0 si no, no se podrá realizar la conexión entre en ordenador y el microprocesador del robot Mindstorm.

Versión	Ancho de banda
Versión 1.2	1 Mbit/s
Versión 2.0 + EDR	3 Mbit/s
Versión 3.0 + HS	24 Mbit/s

Figura 3.6: Tabla de versiones de bluetooth actuales.

CAPÍTULO 4

ARQUITECTURA FUNCIONAL

En este capítulo se expone toda la estructura de la que se compone la parte referente al manejo del robot Mindstorm NXT. Esta capítulo consta de dos módulos, uno estructural donde se mostrarán los diferentes modelos estudiados para el diseño del montaje a realizar con las piezas del Mindstorm y otro computacional donde se encuentra todo el enfoque sobre software que se ha tenido en cuenta a la hora de realizar la programación necesaria para el reconocimiento de los gestos de la mano. También se explicarán todos los módulos programados y su interacción con el fin de hacer funcionar la parte de vision artificial.

4.1. Módulo estructural

Antes de comenzar a exponer la parte computacional de este PFC hay que dejar clara la estructura o modelo que se va a dar al Mindstorm NXT. Puesto que se pueden realizar multitud de montajes diferentes debido a la gran flexibilidad de diseño que se obtiene gracias a la cantidad de piezas de las que se provee en el pack. Se han estudiado tres modelos básicos valorando

su utilidad para la finalidad del proyecto.

4.1.1. Robot con estructura humanoide

Los robots humanoides tratan de imitar el andar humano, para ello utilizan dos piernas articuladas. Habitualmente para el control de este tipo de estructura hacen falta multitud de sensores para detectar la velocidad, ángulo de inclinación lateral y frontal con el fin de poder saber cómo se va desplazando el centro de gravedad y ejecutar los movimientos en respuesta para que haya estabilidad. Puesto que la disponibilidad de sensores está limitada a los que vienen con el Mindstorm NXT esta opción se realiza de manera más sencilla, aumentando considerablemente el área de apoyo de las piernas consiguiéndose así la estabilidad deseada.

Figura 4.1: Robot bípedo con estructura humanoide

4.1.2. Robot con estructura de insecto con múltiples patas

Este tipo de robot cuenta con múltiples patas y tienen una estructura y disposición parecida a la que se puede ver en los insectos. Estas estructuras habitualmente cuenta con seis u ocho patas repartidas por pares de manera simétrica. Lo habitual en este tipo de estructura es que cada pata lleve asociado un único motor, de manera que el control se complica. El bloque

NXT sólo puede controlar tres motores a la vez, por lo que en la estructura resultante para un modelo de seis patas cada motor tiene que controlar dos o tres patas. Ya que la disposición se complica bastante si se disponen los tres motores para sólo controlar dos patas con cada uno, la mejor opción es controlar tres patas por motor y ayudarse de un sistema de engranajes o acoplamiento mecánico para sincronizar los movimientos de éstas.

Figura 4.2: Robot con estructura multipata

4.1.3. Robot con estructura vehículo con ruedas

Es la estructura más sencilla y práctica. El robot adoptaría la forma de vehículo terrestre. La forma de desplazarse también sería la más sencilla, únicamente harían falta dos servomotores con dos ruedas que puedan girar sin limitación. Y una rueda loca para permitir los giros.

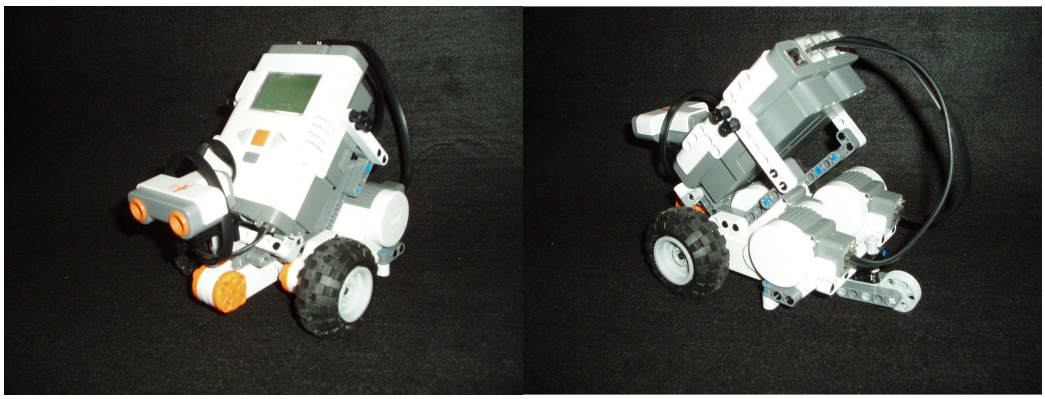

Figura 4.3: Robot con estructura de vehículo

La estructura que se ha adoptado en este proyecto es la última, la de vehículo con ruedas, ya que además de ser la más sencilla permite integrar los movimientos captados por la visión por ordenador de manera simple y ver la reacción casi instantáneamente, es decir, si se muestra el gesto de girar en este modelo se va a apreciar rápidamente como el vehículo empieza a girar, sin embargo, en los otros casos este giro tarda más en ser apreciable.

Por último hay que señalar que se ha dispuesto en la parte frontal del vehículo (sentido de avance del vehículo) un sensor de ultrasonido de manera que el vehículo pueda detectar obstáculos en su camino y frenar para evitar el impacto.

4.2. Módulo computacional

En el módulo computacional se encuentran todas las funciones o programas que se han creado con el fin de lograr la visión por ordenador de los gestos de la mano. Se han definido siete gestos básicos con los que se controla el movimiento del Mindstorm NXT los cuales tienen asociados un pequeño programa para lograr el movimiento deseado. A continuación se explican los comandos a realizar y se muestran las fotografías tomadas por la cámara web para cada uno de los gestos.

- Avanzar (moverse en línea recta). Se activan los dos motores correspondientes (en este proyecto se han utilizado el B y C) con una potencia del 30 por ciento y se manda la señal vía bluetooth al bloque NXT.

Figura 4.4: Gesto de la mano de avanzar

- Frenar (detener el vehículo). Se envía por medio del bluetooth la señal de apagar los motores utilizados al robot. Hay que destacar que al dejar de alimentar al motor esto no hace que se frene inmediatamente, sino que seguirá moviéndose debido a su inercia hasta que el rozamiento lo frene. Esta distancia no es muy grande pero hay que tenerla en cuenta a la hora de la frenada y de regular la distancia de frenada de emergencia con el sensor de ultrasonido.

Figura 4.5: Gesto de la mano de detener el vehículo

- Girar a la izquierda. Ya que los motores son de doble efecto se pueden hacer girar en ambos sentidos, representándose el giro normal como un porcentaje de potencia positivo y el giro contrario un porcentaje de potencia negativo. Se activan los motores a utilizar y para conseguir el giro a la izquierda se da -10 por ciento de potencia al motor de la rueda izquierda y 10 por ciento de potencia al motor de la rueda derecha. Esto produce que el robot gire sobre sí mismo pudiendo hacer giros de cualquier tipo. Una vez asignadas las potencias se envían las señales al Mindstorm.

Figura 4.6: Gesto de la mano de girar a la izquierda

- Girar a la derecha. Idéntico al giro hacia la izquierda salvo que los motores se actúan de forma opuesta, es decir, en este caso el motor izquierdo tiene asignado el 10 por ciento de la potencia y el motor derecho el -10 por ciento lo que crea un giro sobre sí mismo hacia la derecha.

Figura 4.7: Gesto de la mano de girar a la derecha

- Marcha atrás. Igual que el avance pero con potencias negativas, las potencias asociadas a los motores son del -30 por ciento. Al realizar este movimiento hay que tener cuidado, ya que al contrario que en el movimiento de avance, en éste no hay un sensor para evitar impactos.

Figura 4.8: Gesto de la mano de marcha atrás

- Avanzar al doble de velocidad. Igual que el avance pero la potencia de los motores en este caso es del 60 por ciento.

Figura 4.9: Gesto de la mano de avanzar al doble de velocidad

- Salir del programa. Este gesto no lleva asociado un programa sino que sirve para dejar de ejecutar el programa de visión por ordenador en tiempo real.

Figura 4.10: Gesto de la mano de salir del programa

Para realizar la visión artificial por ordenador se han analizado los cuatro reconocedores de patrones más conocidos. A continuación se hará una breve explicación de cada uno de los métodos y los fundamentos en los que se basa para el reconocimiento de patrones. Finalmente se realizará una valoración de cada método explicando el método seleccionado. Estos cuatro reconocedores son: *template matching* o correspondencia entre patrones, reconocimiento estadístico, reconocimiento sintáctico o correspondencia estructural y redes neuronales.

4.2.1. *Template matching*

Para este tipo de reconocimiento es necesario disponer de una plantilla (*template*) o un patrón prototipo. Dependiendo de la plantilla se puede realizar el reconocimiento de dos maneras diferenciadas. En primer lugar, reconocimiento basado en características, en donde sólo se buscarán características determinadas dentro de las imágenes, sin darle importancia al resto de la imagen. La segunda opción es el reconocimiento basado en plantilla, donde se busca una similitud con el total de la plantilla importando todo el conjunto.

Hay que destacar que este método no busca únicamente una coincidencia exacta de las características, sino que busca similaridades entre la plantilla y la imagen donde se pretenden encontrar los patrones. El grado de similitud que se quiera respecto a la plantilla condicionará la eficiencia de este método pudiendo crear falsos positivos teniendo demasiado margen o no detectar un patrón correcto por pedir excesiva similitud.

4.2.2. Reconocimiento estadístico

Para llevar a cabo este reconocimiento es necesario inicialmente la creación de un vector de n elementos, en el que cada elemento representa una característica. De esta manera, cada patrón es respresentado por un vector n-dimensional.

El procedimiento trata de seleccionar características asignando vectores patrones para diferenciar categorías que ocuparan regiones compactas o separadas en el espacio n-dimensional.

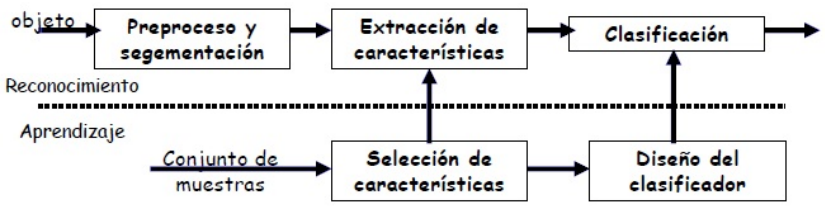

Figura 4.11: Esquema de funcionamiento de un reconocedor estadístico.

4.2.3. Reconocimiento sintáctico o correspondencia estructural

Este método se basa en que cada patrón esta compuesto a su vez por subpatrones elementales, los cuales se pueden considerar como los elementos básicos con los que se puede construir cualquier elemento. De esta manera, todo patrón complejo puede verse como un conjunto de subpatrones.

El objetivo es encontrar las relaciones estructurales que guardan los objetos de estudio, utilizando la teoría de lenguajer formales, teoría de autómatas, etc. Para finalmente poder contruir una gramática que describa la estructura del universo de objetos, pudiendo comparar las estructuras del patrón con las de cualquier imagen.

4.2.4. Redes neuronales

Las redes neuronales artificiales son modelos que intentan reproducir el comportamiento del cerebro humano o animal. Como tal modelo, realiza una simplificación, averiguando cuáles son los elementos relevantes del sistema, bien porque la cantidad de información de que se dispone es excesiva o bien porque es redundante. Una elección adecuada de sus características, más una estructura conveniente, es el procedimiento convencional utilizado para construir redes capaces de realizar determinadas tareas. Una red neuronal se compone de unidades llamadas neuronas. Existen tres tipos de unidades en cualquier sistema: entradas, salidas y ocultas. Las unidades de entrada reciben señales desde el entorno, las de salida envían la señal fuera de la red y las unidades ocultas son aquellas cuyas entradas y salidas se encuentran dentro del sistema. Cada neurona recibe una serie de entradas a través de interconexiones y emite una salida. Esta salida viene dada por tres funciones: 1. Una función de propagación, que por lo general consiste en el sumatorio de cada entrada multiplicada por el peso de su interconexión (valor relativo que se asigna a cada elemento de entrada). Si el peso es positivo, la conexión se denomina *excitatoria* y si es negativo, se denomina *inhibitoria*. 2. Una función de activación. Se requiere una regla que combine las entradas con el estado actual de la neurona para producir un nuevo estado de activación. Esta función produce un nuevo estado de activación en una neurona a partir del estado que existía y la combinación de las entradas con los pesos de las conexiones. Puede no existir, siendo en este caso la salida la misma función de

propagación. 3. Una función de transferencia, que se aplica al valor devuelto por la función de activación. Se utiliza para acotar la salida de la neurona y generalmente viene dada por la interpretación que queramos darle a dichas salidas. Algunas de las más utilizadas son la función sigmoidea (para obtener valores en el intervalo [0,1]) y la tangente hiperbólica (para obtener valores en el intervalo [-1,1]). A continuación se muestra un esquema de un modelo con múltiples neuronas de entrada y una única neurona de salida.

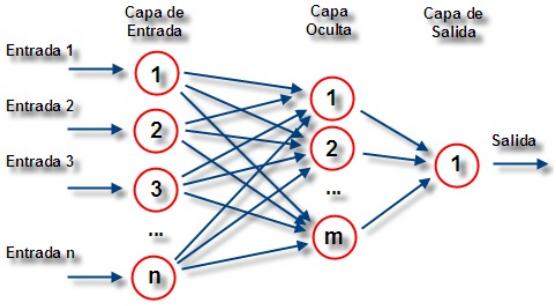

Figura 4.12: Red neuronal artificial con n neuronas de entrada, m neuronas en su capa oculta y una neurona de salida

Para diseñar una red neuronal hace falta una selección correcta de los elementos a estudiar y aparte es necesaria la realización de un entrenamiento para ajustar los pesos de la función de propagación para conseguir la salida deseada ante cada situación. El uso de una red neuronal puede suponer algunas ventajas:

- Tolerancia a fallos: Aunque se dañe parcialmente la red neuronal gracias a la información redundante que se recibe se puede seguir respondiendo
- Tiempo real: La estructura de una red neuronal artificial es paralela, por lo que si es implementada con ordenadores se pueden obtener respuestas en tiempo real.
- Flexibilidad: Se pueden manejar cambios no importantes en la información de entrada, como señales con ruido u otros cambios en la entrada (por ejemplo si la información de entrada es la imagen de un objeto, la respuesta correspondiente no sufre cambios si la imagen cambia un poco su brillo o el objeto cambia ligeramente).

4.2.5. Análisis de modelos y selección

En cuanto al método del *template matching* se encuentran principalmente dos desventajas. La primera es el alto coste computacional necesario para realizar este método, ya que las comparativas requieren profundizar en la estructura de la imagen a tratar aunque, en la actualidad, con el avance de los procesadores esta desventaja no supone un factor determinante. El segundo problema es la rigidez de las plantillas, lo que genera problemas cuando aparecen las imágenes distorsionadas debido, por ejemplo, a la iluminación o a la orientación.

El reconocedor estadístico supone que se tiene un conjunto de medidas numéricas con distribuciones de probabilidades conocidas con las cuales se realiza el reconocimiento. Otro factor que influye a la hora de este método es la capacidad para separar los patrones en diferentes clases, ya que una mala elección reducirá la eficiencia de este reconocedor.

El método de reconocimiento sintáctico se basa en la creación de la gramática con la que puede analizar las imágenes a tratar. Esta creación es relativamente compleja y lleva asociada un entrenamiento a partir de un conjunto de muestras.

Al realizar una red neuronal bien entrenada para el caso de estudio se podrían conseguir una serie de características como área de la mano, número de agujeros, perímetro de la mano, curvatura, número de dedos, circularidad y ángulo de inclinación. Muchas de esas características resultarían inservibles debido a la simplicidad de los gestos seleccionados para los movimientos del Mindstorm NXT. Teniendo en cuenta la relativa complejidad que hace falta para la realización de la red neuronal, su etapa de aprendizaje y que no resulta tan fiable cuando se sufre cambios en el entorno, como podría ser el fondo de la imagen, además de que se obtendría mucha información adicional no necesaría se considera que este método no es una buena opción.

Una vez analizados todas las opciones se ha optado por tomar como referencia el método *template matching* pero ajustándolo a las necesidades que requiere este proyecto. Como ya se comentó anteriormente la capacidad computacional en principio no es un problema gracias a los procesadores actuales y en cuanto al problema de rigidez de la plantilla tampoco será un inconveniente ya que las muestras a analizar se realizan en un entorno controlado, del cual se extraerán las plantillas y a su vez, a posterior, las imágenes

en tiempo real que se desean analizar.

Para crear el reconoceder de patrones se han realizado cuatro módulos diferenciados que interactúan unos con otros con la finalidad de que se reconozcan los gestos definidos para la aplicación. Estos cuatro módulos son los siguientes: inicialización, procesado de imágenes, contador de dedos y el programa principal. Con la finalidad de poder interpretar mejor los procedimientos que se realizan en estos módulos conviene definir varias operaciones que se van a utilizar.

- Umbralización: A este proceso también se le pude llamar binarización, ya que la finalidad es transformar una imagen en escala de grises a otra en la que todos los elementos sean blanco o negro. En esta operación se analiza cada elemento de una matriz comparándolo con un umbral, T. Si el valor del elemento es mayor al umbral se asignará el máximo valor y si es menor o igual al umbral se le asignará el menor valor posible. El rango de escala de grises va desde 0 hasta 255, significando el 0 negro y 255 el blanco. De forma matemática se puede expresar de la siguiente manera:

$$f(x,y) = \begin{cases} 255 & f(x,y) > T, \\ 0 & f(x,y) \leq T. \end{cases}$$

  donde:

  $f(x,y)$: Valor numérico del elemento a umbralizar dentro de la matriz.

  T: Nivel de umbral seleccionado expresado numéricamente.

- Reescalado: Consiste en modificar las dimensiones de algún objeto, en este caso una matriz representativa de una imagen. El proceso se realiza sustituyendo grupos de elementos por uno que represente a la mayoría, es decir, si se escala una matriz al 50 por ciento significa que se va a reducir su tamaño entre cuatro, ya que reducirá a la mitad tanto el eje vertical (y) como el eje horizontal (x). De esta manera se escogerían cuatro elementos contiguos (matriz de 4 elementos, dos filas y dos columnas) y se sustituirían por uno, representando este a la mayoría. En caso de igualdad de elementos de ambos colores se ha de establecer que decisión tomar. A continuación se muestra una figura en la que se muestra gráficamente la toma de decisiones del reescalada para el ejemplo comentado.

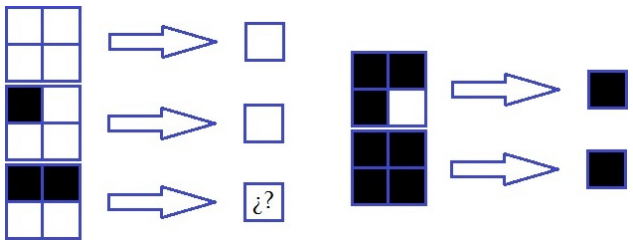

Figura 4.13: Ejemplo de reescalado de 4 elementos.

- Dilatación: En una imagen binarizada (en blanco y negro) se elegirá si se desea hacer este proceso al blanco o al negro. Una vez elegido, por ejemplo el blanco, en las zonas en las que haya contacto de elementos de ambos colores la primera fila de elementos negros pasarán a ser blancos, es decir, todo elemento negro en contacto con un elemento blando en la imagen inicial se convertirán en blancos.

- Erosión: Es la operación inversa a la dilatación, de manera que para el ejemplo anterior si se selecciona el blanco se transformarán en elementos negros todos aquellos que estuvieran inicialmente en contacto con elementos negros.

- *Opening* y *Closing*: Con estos nombres se conocen a dos operaciones morfológicas en el tratamiendo de imágenes que consisten en realizar dos procesos de forma consecutiva, en el caso del *opening* un erosión seguida de una dilatación y en el caso del *closing* de una dilatación seguida de una erosión. Estas operaciones habitualmente son utilizadas para la eliminación de ruido en el tratamiento de imágenes, además de suavizar las fronteras.

Inicialización

El primer paso y con la finalidad de poder adaptar el sistema para la utilización de cualquier usuario se toman unas fotografías a modo de patrón de forma que se pueda obtener información como el largo de la mano con los dedos estirados o de las dimensiones del puño. En este apartado se realizan cuatro fotografías y para hacerlo más intuitivo para el usuario se realizan siguiendo las instrucciones que se dan mediante ventanas emergentes. Las ventanas que van a aparecer (de manera secuencial) se muestran en la siguiente imagen.

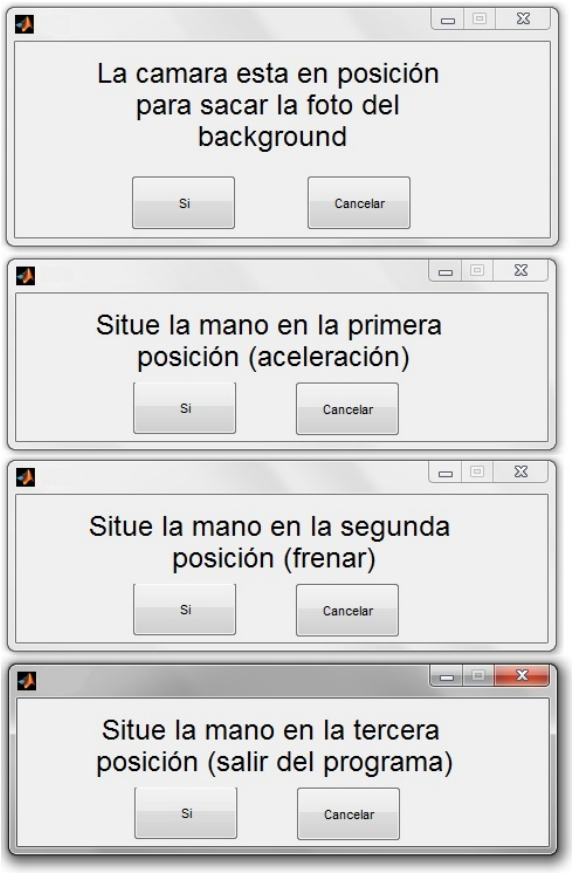

Figura 4.14: Ventanas emergentes que aparecen durante el programa de inicialización

La primera ventana emergente en aparecer es la que saca la foto del fondo (*background*). Al pulsar el botón 'sí' se conecta con la cámara web y se comprueba que la conexión se ha realizado correctamente. A continuación se abre una ventana en la que se visualiza las imágenes que va captando la cámara web, se esperan cinco segundos para que se estabilice la imagen ya que al iniciarse la cámara tiene que detectar la luminosidad del entorno y ajustarse automáticamente. Tras este ajuste se realizará la foto del fondo que será usada a partir de ese momento, en caso de variación en el fondo se tendrá que repetir al menos este paso.

En la segunda y la tercera ventana el proceso que se realiza es el mismo, se saca la foto en la posición que se pide y se ejecuta el programa de procesado de la imagen, guardándose la imagen resultante para futuras consultas de

dimensiones. A continuación se muestra un ejemplo de como se ve la pantalla durante esta inicialización.

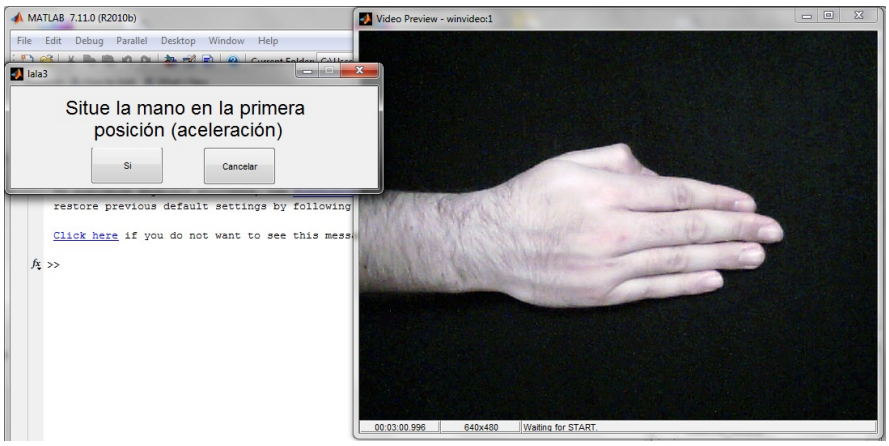

Figura 4.15: Ejemplo de visualización de la pantalla en la segunda ventana

En la cuarta y última ventana el proceso será similar, se saca la foto y se ejecuta el programa de procesamiento. Una vez terminado se muestran por pantalla las fotos tratadas de las tres posiciones solicitadas y sus dimensiones. Tras esto se calcula el número de píxeles blancos que hay en el borde de la imagen (la muñeca o antebrazo) ya que más adelante se usará este número como referencia para distinguir entre cuando hay un solo dedo y la posición de avanzar.

Una vez terminado este paso se procede a ejecutar el programa principal el cual irá analizando las imágenes en tiempo real y ejecutando los comandos que correspondan en función de las imágenes procesadas. Antes de entrar a explicar la toma de decisiones que se realiza en el programa principal se debe conocer el proceso de tratamiento de imágenes y el proceso de cuenta de dedos ya que éstos proveen de los datos necesarios para decidir qué acción se lleva a cabo.

Procesado de imágenes

Este es el bloque en el que se realiza el tratamiento de las imágenes obtenidas mediante la cámara web y se dejan preparadas para realizar su interpretación por el programa principal. Este procesamiento cuenta con una

serie de pasos con los que se va adecuando la imagen original a la imagen final que se busca.

Del apartado anterior de inicialización ya tenemos en memoria la foto del fondo, para realizar el tratamiento se tiene que adquirir otra fotografía donde se vea la mano con el gesto del movimiento que se desea realizar por el Mindstorm NXT. Esta adquisición se realiza mediante el programa principal que se explica en un apartado posterior.

A continuación se muestran la fotografía del fondo y un ejemplo de fotografía en tiempo real, donde cabe destacar que se ha introducido un objeto (un bolígrafo) en el fondo para ver más claramente el tratamiento de la imagen. Puesto que estas imágenes son las originales tomadas por la cámara web al almacenarlas en memoria lo hacen en formato RGB ('*red green blue*' o 'rojo verde azul'), el cual guardará una matriz tridimensional que consta de tres matrices con la misma imagen pero donde sólo se tienen en cuenta los valores de los puntos en función de un filtro de color, es decir, el primer nivel tendrá los valores desde el punto de vista del color rojo, el segundo desde el verde y el tercero desde el azul. De esta manera se tiene una matriz de 640x480x3 puntos.

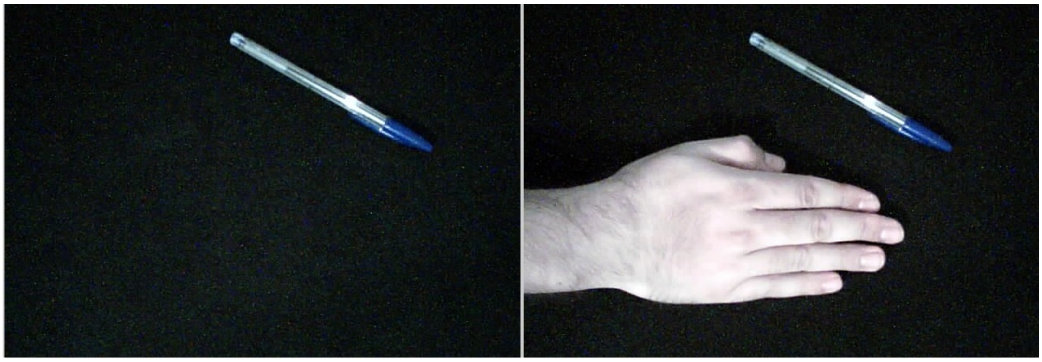

Figura 4.16: Fondo utilizado para la adquisición de imágenes (izquierda) y fotografía tomada con el gesto de avance del robot (derecha) en formato RGB.

Ya que manejar matrices tridimensionales ralentiza mucho el procesado, ya que se tiene que analizar el triple de elementos, el primer paso que se ha de realizar es la transformación de estas imágenes desde el formato RGB a un formato de escala de grises, o lo que es lo mismo, cambiamos la matriz tridimensional por una matriz unidimensional. Esta transformación se

puede realizar de manera automática en Matlab con el comando *RGB2gray*, obteniendo una imagen en 256 niveles de grises, indicando el 0 el gris más oscuro (negro) y el 255 el más claro (blanco), de esta manera se obtienen las siguientes imágenes.

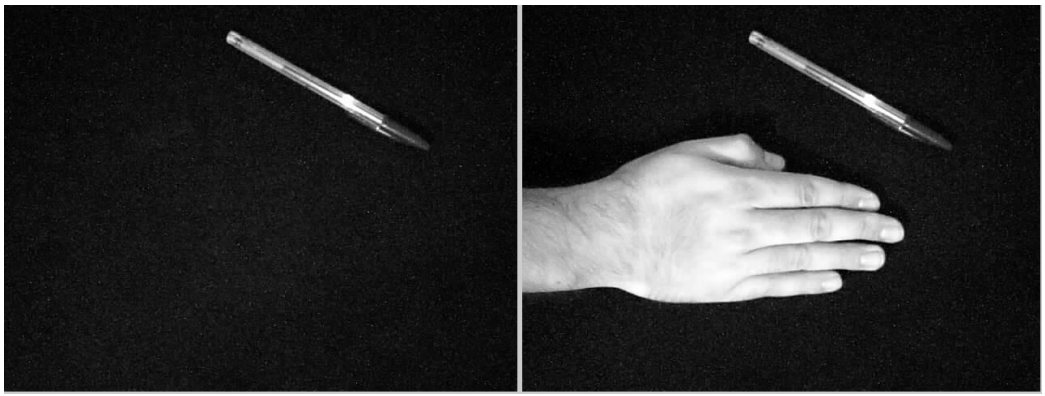

Figura 4.17: Fondo utilizado para la adquisición de imágenes (izquierda) y fotografía tomada con el gesto de avance del robot (derecha) en escala de grises.

Llegados a este punto hay que destacar que esta escala de grises se almacena en una matriz de 640x480 puntos donde cada punto va desde un valor de 0 (negro) a un valor de 255 (blanco). El siguiente paso a realizar es la resta de la imagen adquirida menos el fondo, aunque esto no siempre debe de ser así puesto que en el caso de tener un fondo claro (los puntos son valores altos) al restárselo a la imagen adquirida se tendería a saturar la imagen ya que esta matriz no puede tener valores negativos por lo que si en la resta un valor va a ser negativo se queda como un cero. En ese otro caso habría que restar el fondo menos la imagen adquirida. Para el estudio del caso presente y puesto que se dispone de un fondo negro se realiza la resta de la imagen adquirida menos el fondo.

Hasta este punto las imágenes tienen unas dimensiones de 640 por 480 puntos pero ya que más adelante se va a tener que recorrer las matrices para hacer diferentes operaciones en tiempo real y con la finalidad de no necesitar una capacidad computacional muy elevada se ha decidido redimensionar la imagen con una escala de 0,5 con lo que se consigue una imagen de 320 por 240 puntos lo cual reduce por cuatro el número total de puntos de la matriz.

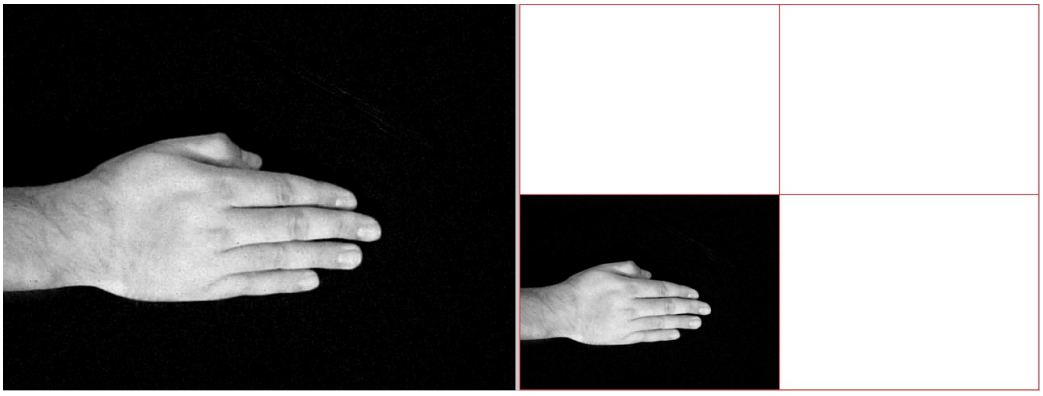

Figura 4.18: Imagen resultante de la resta entre la imagen y el fondo a la izquierda. A la derecha la redimensión con escala de 0,5.

Una vez se llega a este punto se tiene la imagen preparada para seguir su tratamiento. El primer paso a realizar es una binarización de la imagen de manera que sólo se obtengan puntos blancos o negros. Para realizar esta binarización es necesario establecer un umbral con el cual se decide cuando un punto se considera negro o blanco. Para el caso estudiado, con el fondo negro, se ha obtenido experimental buenos resultados con un valor de 45, de manera que si el valor del punto de la matriz es mayor a 45 se le asigne un valor de 255 (blanco) y en caso contrario se le asigne un 0 (negro).

Cabe destacar que también se estudió el caso con el fondo blanco y el umbral resulta un número muy similar teniendo en cuenta que para ese caso hay que realizar la resta del fondo y de la imagen de forma contraria al caso estudiado, es decir, hay que restarle al fondo la imagen que se esté procesando.

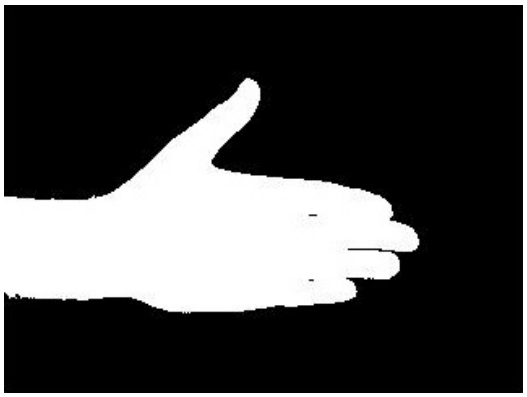

Figura 4.19: Imagen binarizada con un umbral de 45.

En este punto la imagen esta binarizada pero con el fin de arreglar pequeños errores que se pueden producir en la binarización se suele hacer un proceso de dilatación-erosión (*closing*) el cual consiste en ampliar la frontera de uno de los dos colores y a continuación reducirlo. Este proceso se realiza para eliminar pequeños fallos o simplemente para suavizar las fronteras entre ambos colores de manera que se eliminan los bordes rugosos. En el caso de estudio con el fondo negro si tan sólo se realiza un llenado de agujeros (eliminación de las islas) de la zona blanca bastaría como se aprecia en la siguiente imagen.

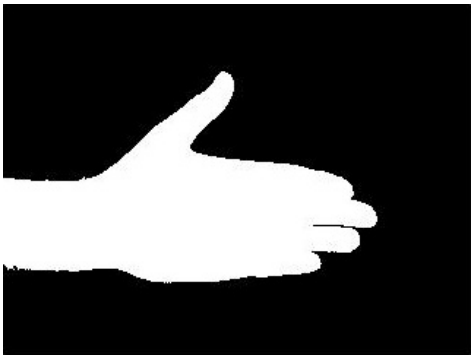

Figura 4.20: Imagen resultante del llenado de agujeros.

A continuación se le realizan dos procesos de erosión. En este caso no es tan claro la realización de este paso, sin embargo, en el caso de tener un fondo claro las fronteras quedan menos definidas por culpa de las sombras que producen los dedos, ya que al ser oscuras normalmente parte de ellas pasarán el umbral y se incluirán en la binarización. Para este caso los procesos de erosión quedarán como se muestra en las siguientes figuras.

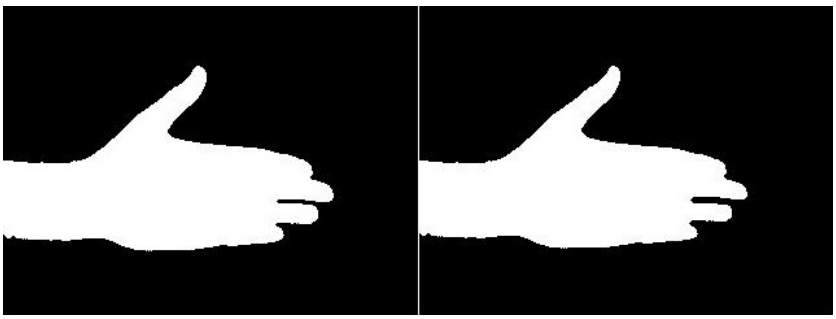

Figura 4.21: Imagen resultante de un proceso de erosión (izquierda) y con un segundo proceso de erosión (derecha).

Una vez se tiene la imagen lista, se le realiza una última operación con la finalidad de sólo mantener información útil dentro de la matriz, recortándose toda la imagen que no da ningún tipo de información. Este proceso se realiza recorriendo la matriz por los tres bordes de la imagen en los que se desconoce hasta donde llega la parte blanca que es la que contiene la información. Estos tres bordes son el superior, el derecho y el inferior, ya que el borde izquierda es donde se encuentra el antebrazo. Por ejemplo, se empieza a recorrer desde la parte superior y horizontalmente punto por punto hasta que se detecta un punto blanco, ese es el límite superior de la imagen que se desea. Al realizar esta operación también en el lado derecho e inferior se consigue una imagen recortada de la que se puede obtener datos como el largo del gesto en la imagen estudiada, pudiéndose hacer una comparación con las imágenes obtenidas en el proceso de inicialización.

Figura 4.22: Imagen final recortada.

Contador de dedos

Este módulo se ejecuta una vez el programa principal ha comprobado que la señal no coincide ni con el gesto de frenado ni con el de salida del programa, es decir, se ejecuta cuando se tiene la certeza de que la mano está estirada y hay que comprobar el número de dedos. Ya que la posición de la mano en la imagen que se va a recibir es conocida se sabe donde van a estar los dedos. Por otra parte de la fase de inicialización se puede extraer el rango de distancia o píxeles en el que se van a encontrar los dedos. Teniendo en cuenta que se conoce el ancho de la imagen del gesto de frenada (puño cerrado) y el ancho de la imagen del gesto de avance (palma estirada) se puede ver cuál es el rango en que se van a encontrar los dedos. A continuación se muestra una figura en la que se aprecia el rango de distancias (respecto al ancho de la imagen) en el que se van a encontrar los dedos.

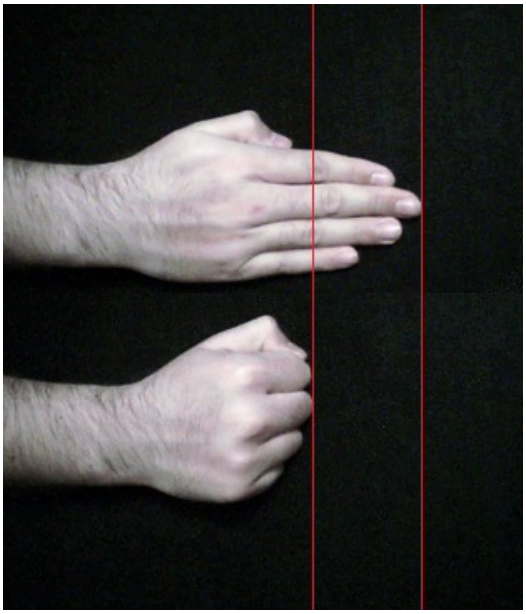

Figura 4.23: Rango donde se encuentran los dedos.

Hay que destacar que a este módulo las imágenes que llegan en tiempo real ya han sido procesadas por lo que están recortadas y binarizadas, de manera que para contar los dedos que hay tan sólo hay que analizar una línea vertical en el rango donde se sitúan los dedos, pero teniendo cuidado de no perder datos por haber seleccionado una línea vertical en la que falte información como se muestra en la siguiente figura.

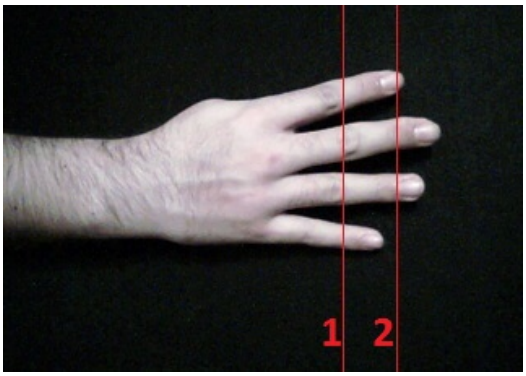

Figura 4.24: Líneas verticales donde se comprueba el número de dedos.

Como se puede apreciar en la figura anterior si se realiza la cuenta en la

línea 1 se obtiene toda la información, sin embargo, si se cuenta en la línea 2 se omite la información del meñique. Para que el sistema sea más robusto a este tipo de errores se realiza dos veces el proceso de conteo contrastando los dos resultados.

Teniendo en cuenta que es conocido el rango en el que se encuentran los dedos y que se realizan dos conteos diferentes lo primero que se hace es establecer las distancias en las que se cuentan los dedos. Finalmente para establecer estas distancias se ha optado por dividir en tres verticalmente el rango y realizar los conteos en la primera y segunda separación, de manera que los conteos se realizan en la distancia del ancho de la imagen de frenada (A) más un tercio (B) o dos tercios del rango (C). En la siguiente figura se muestran las distancias a tener en cuenta.

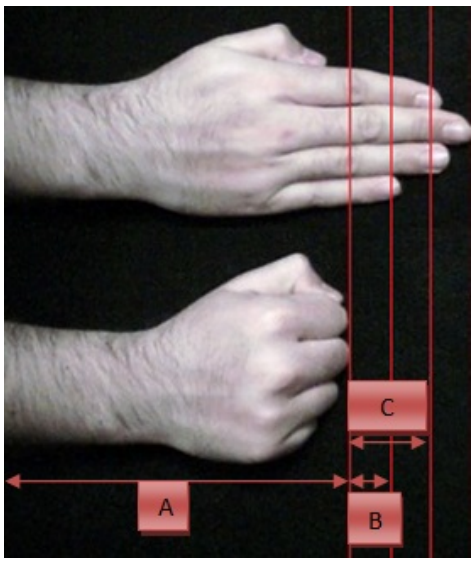

Figura 4.25: Distancias a tener en consideración en el conteo de dedos.

Como ya se comentó, a este programa llegan las imágenes binarizadas de manera que para contar los dedos tan sólo hay que ver cuántos cambios de color (de blanco a negro y de negro a blanco) hay en la vertical, dividir ese número por dos y redondearlo al alza. Hay que considerar que cuando solo hay tres cambios de color, o lo que es lo mismo, un sólo dedo hay que comprobar que sea un dedo y no la posición de avanzar. Esto se consigue gracias a un valor que se obtiene en el módulo de inicialización donde se cuenta el número de pixeles que hay en el antebrazo. Al haber un sólo dedo se cuentan los píxeles de ancho que tiene y se compara con la mitad del número de píxeles

del antebrazo, si es mayor se considera que no hay dedos sino la palma entera y si es menor se considera que hay un dedo. Este procedimiento se realiza solamente en la primera comprobación, ya que si en esta se obtiene que se encuentra la palma de la mano, la segunda comprobación va a ser siempre igual. En caso de no ser cero dedos se procede a la segunda comprobación realizándose un nuevo conteo. Una vez se han realizado los dos conteos se devolverá al programa principal el mayor valor de las dos comprobaciones ya que la información que se pierde en este caso siempre será contar de menos así que la cuenta correcta será la mayor.

Programa principal

El último de los módulos que componen el programa para la visión artificial es el programa principal que es el encargado de obtener las imágenes, mandarlas a procesar, mandar a contar dedos si es necesario y ejecutar los comandos correspondientes en función de toda la información recibida de los módulos ejecutados en tiempo real.

Hay que comentar que este bloque no tiene contacto directo con el módulo de inicialización pero sí que utiliza la información recopilada en aquel módulo, que es principalmente las dimensiones de las imágenes. Especialmente importantes son dos de esas dimensiones ya que hacen únicos dos gestos tan sólo con una dimensión, el ancho de la imagen de frenado ya que es el único gesto en el que no se encuentran estirados ninguno de los dedos y el alto de la imagen de salir del programa ya que al estirar el pulgar esta medida es mucho mayor a cualquier otra en los gestos propuestos.

Este módulo consta principalmente de un bucle el cual se estará ejecutando continuamente hasta que se muestre el gesto de salir del programa. Dentro de este bucle y con el objetivo de minimizar el envío de información redundante al Mindstorm NXT se crean dos variables que se irán actualizando en cada bucle, de manera que siempre se sepa el estado en el que se encuentre el programa. Se pueden distinguir siete estados, uno por cada gesto. Antes de entrar al bucle se ejecutan dos sentencias, la primera que se ejecuta es asignar como estado anterior que el Mindstorm se encuentra frenado y la segunda es la línea que activa el ultrasonido para que vaya detectando si hay obstáculos en su camino a recorrer y que en caso de peligro de colisión se ejecute una frenada de emergencia.

Una vez se entra en el bucle y teniendo en cuenta que la cámara web ya está funcionando (esta activa desde la inicialización) lo primero que se hace es tomar una fotografía la cual se manda al módulo de procesado de imágenes. Una vez se termina el procesamiento lo siguiente que se hace es ver las dimensiones finales de la imagen resultante, pudiéndose saber si el gesto es el de salida del programa el cual al ser prioritario es el primer gesto que se comprueba. Si el alto no coincide con el de la salida del programa obtenida en la inicialización más un margen (en este caso se toman 10 píxeles de más o de menos a la dimensión original) se prosigue a hacer la siguiente comprobación que es ver si es el gesto de frenada. Para este gesto al igual que en el de salida del programa sólo se comprueba una dimensión, la del ancho de la imagen. Como ya se comentó anteriormente, el resto de gestos tienen al menos un dedo estirado haciendo que esta medida sea única para este gesto. Al igual que en la comprobación anterior se compara este ancho con el obtenido en la inicialización más un margen. En este caso este margen hay que estudiarlo ya que si la imagen tiene menos píxeles está claro que será el gesto de frenada pero en caso de tener más no se sabe si simplemente se está abriendo la mano para realizar otro gesto o si hay algún dedo estirado.

Como ya se explicó en el módulo anterior del contador de dedos se hacen dos comprobaciones al realizar el conteo de dedos y estas son a distancias fijas no se puede permitir que el programa crea que hay dedos debido al ancho de la imagen cuando esta dimensión es menor a la de la segunda comprobación. De esta manera el margen para esta comprobación es de 10 píxeles de menos y dos tercios de la diferencia entre el puño y la palma de la mano estirada por encima. En caso de que la medida del ancho de la imagen tratada en tiempo real esté dentro de este margen se verifica si el estado anterior es de frenada, sin realizar ningún comando si es cierto y activando el comando de frenada en caso de ser falso. Si la dimensión del ancho supera ese valor se considera que la imagen posee información en forma de dedos procediéndose a ejecutar el módulo del contador de dedos. Una vez se tiene la respuesta de cuántos dedos se detectan en la imagen en función de ese número se hacen las comprobaciones y se ejecutando los comandos necesarios. A continuación se muestran estas comprobaciones y comandos a ejecutar en función del número de dedos.

- Número de dedos igual a 0. Esto significa que la imagen que se está estudiando no tiene dedos separados pero que la palma esta estirada, es decir, que el gesto es el de avance del Mindstorm NXT. Una vez se sabe que este es el gesto que se está estudiando se debe realizar la comproba-

ción de si hay obstáculos a una distancia menor de 10 centímetros. En caso de haberlos se comprueba si el estado anterior ya es frenado y si no lo es se frena. En caso de no haber obstáculos cercanos se comprueba si el estado anterior es avanzando sin realizarse ningún comando o si era otro se ejecuta el comando de avance del Mindstorm NXT.

- Número de dedos igual a 1. Si sólo se detecta un dedo y ya se descarta que sea el gesto de avance; esto quiere decir que este gesto es el de girar a la izquierda. Simplemente se verifica si el estado anterior era girando a la izquierda y en caso contrario se ejecuta el comando de giro a la izquierda.

- Número de dedos igual a 2. En este caso se quiere girar a la derecha. Se comprueba si el estado anterior era el giro deseado y en caso contrario se ejecuta el comando de giro a la derecha.

- Número de dedos igual a 3. Este gesto supone que se dé marcha atrás al Mindstorm NXT. Se verifica que el estado anterior no fuera ya dando marcha atrás y en caso contrario se ejecuta el comando pertinente para realizar esa acción.

- Número de dedos igual a 4. Con este gesto se pretende que el Mindstorm avance así que al igual que en el caso de no haber dedos se debe de hacer la comprobación de si hay obstáculos. En caso de haberlos se comprueba si ya se está parado y si no se ejecutará el comando de frenada. En caso de no haber obstáculos se comprueba si el estado anterior era avanzando al doble de la velocidad normal, sin realizarse ningún comando en caso positivo y activando el avance rápido en caso contrario.

Finalmente para terminar el bucle se actualiza el estado anterior con el estado actual, que queda registrado al decidirse de que gesto se trata en la imagen en tiempo real. Aparte de esto en caso de que el estado anterior y el actual sean el mismo se aumenta la cuenta de un contador que indica que se está reiterando un gesto. En caso de que los estados no coincidan este contador se reinicia poniéndose a cero. Este contador tiene la función de hacer más suave los cambios de comando ya que si, por ejemplo, se está en la posición de frenada y se quiere pasar a la de avance en la transición del movimiento de los dedos igual se realiza el bucle varias veces detectándose cada vez un comando diferente mandándose un comando a realizar por el Mindstorm NXT por cada iteración lo que hace que el robot vaya haciendo movimientos no deseados y dando la sensación de ir dando saltos. Experimentalmente

se ha comprobado que con que se repita tres veces un comando se consigue eliminar este problema de forma considerable. Si se aumentara este número de repeticiones se conseguiría eliminar este problema el todo pero al tener que esperar más ciclos para dar por bueno el comando a ejecutar quiere decir que el robot continuará ejecutando el comando anterior mientras ya se está haciendo el gesto nuevo, es decir, se introduce un desfase entre el gesto y la transmisión del comando pertinente al Mindstorm NXT.

Una vez se salga del bucle porque se haya detectado y procesado el gesto de salir del programa se ejecuta una última sentencia en el que se manda al Mindstorm NXT la señal de frenada ya que al salir del programa se desconoce en qué estado queda el robot y al no haber un comando futuro no importa reiterar el comando en el caso de que ya estuviera parado.

CAPÍTULO 5

EXPERIMENTACIÓN

En este capítulo se intentará exponer los resultados y conclusiones obtenidas mediante la experimentación realizada para la comprobación del programa. Dentro de estos resultados se mostrarán los tiempos de ejecución de los principales programas y acciones, pudiéndose estimar los tiempos de reacción del robot Mindstorm. Finalmente se realizará la exposición gráfica de la simulación de movimientos.

Los datos extraídos de las pruebas serán separados en tres apartados: visión artificial por computador donde se verá el tratamiento de imágenes y el contador de dedos, controlabilidad donde se hablará de los tiempo de ejecución y movilidad donde se expondrá mediante secuencias de fotogramas ejemplos de movimientos.

5.1. Visión artificial

En esta sección se muestra el resultado final que se obtiene del tratamiento de la imagen que se realiza antes de ejecutar el módulo que decide cual es el

comando adecuado al gesto realizado.

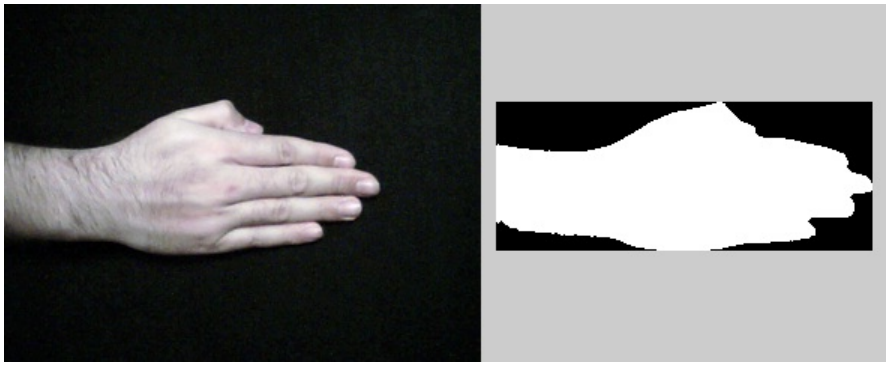

Figura 5.1: Imagen original y resultado del tratamiento de imagen: avanzar.

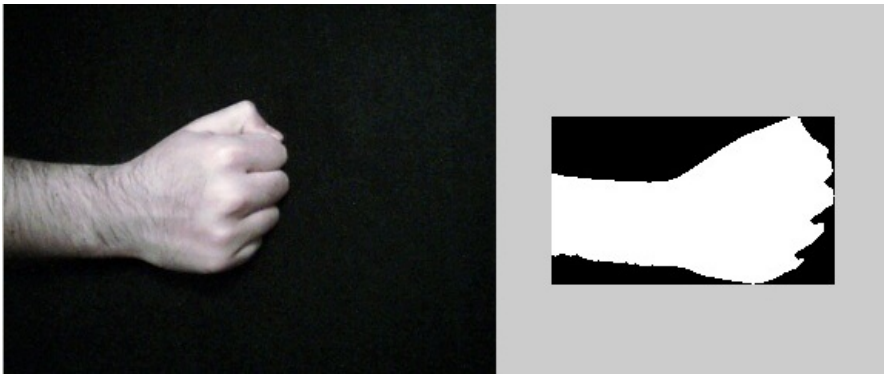

Figura 5.2: Imagen original y resultado del tratamiento de imagen: frenar.

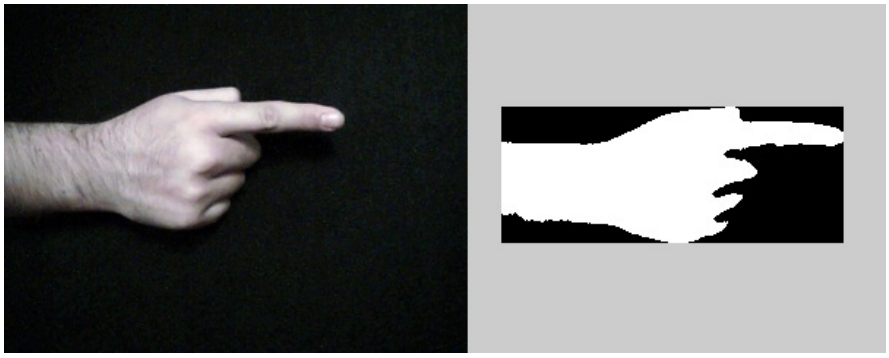

Figura 5.3: Imagen original y resultado del tratamiento de imagen: girar a la izquierda.

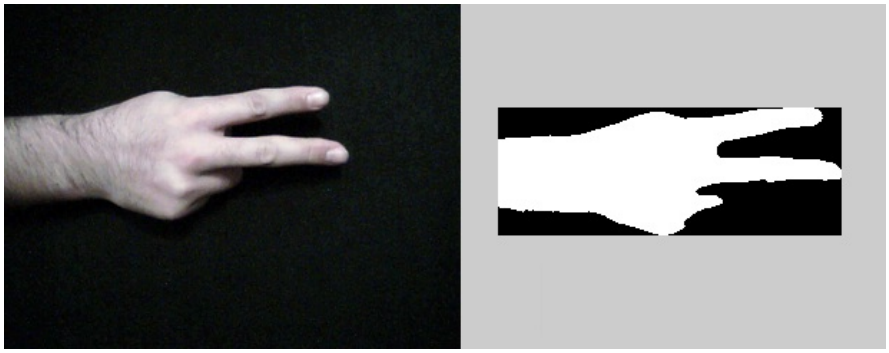

Figura 5.4: Imagen original y resultado del tratamiento de imagen: girar a la derecha.

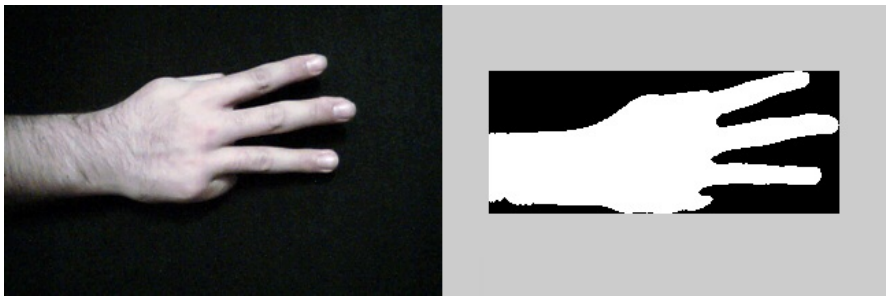

Figura 5.5: Imagen original y resultado del tratamiento de imagen: marcha atrás.

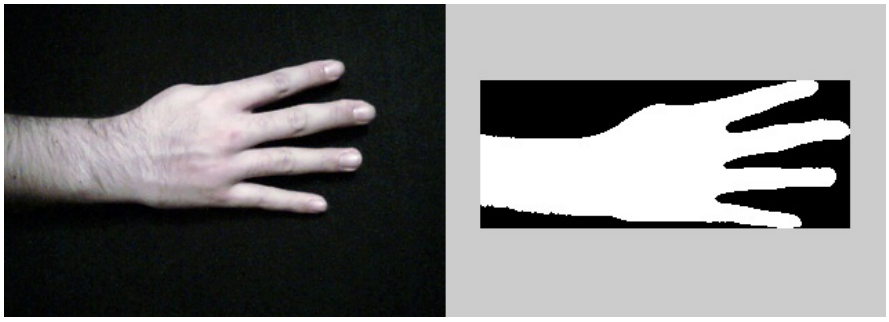

Figura 5.6: Imagen original y resultado del tratamiento de imagen: avanzar rápido.

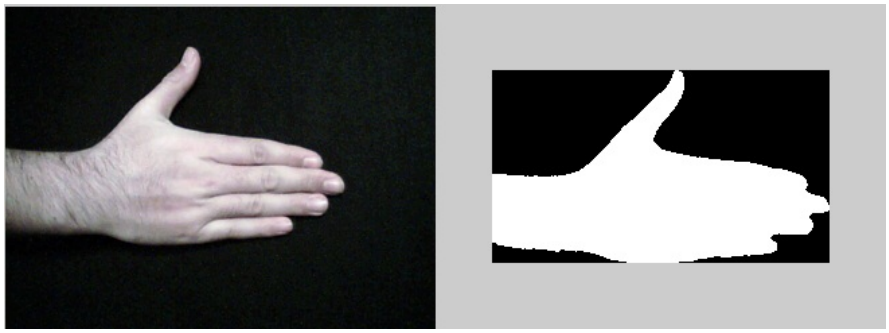

Figura 5.7: Imagen original y resultado del tratamiento de imagen: salir del programa.

A partir de estas imágenes tratadas el programa principal decide si ya tiene suficiente información o si es necesario ejecutar el módulo de contador de dedos para tener todos los datos necesarios para la toma de decisiones. En el caso en que sea necesario ejecutar el módulo del contador de dedos el programa analizará las imágenes como ya se comentó anteriormente dando los siguientes resultados.

```
>> [numero_dedos_comprobacion1 numero_dedos_comprobacion2 numero_dedos]

ans =

     1     1     1
```

Figura 5.8: Ejemplo 1 de contador de dedos.

Como ya se explico en el capítulo 4 el ancho de dedo está definido en función del ancho del antebrazo, pudiéndose normalmente utilizar dos dedos juntos como si fuera uno sólo. Esto es una ventaja para ciertos cambios de movimiento, como por ejemplo para pasar del movimiento de avance al giro a la derecha ya que separando el índice-corazón y el anular-meñique se conseguiría el movimiento.

```
>> [numero_dedos_comprobacion1 numero_dedos_comprobacion2 numero_dedos]

ans =

     2     2     2
```

Figura 5.9: Ejemplo 2 de contador de dedos.

```
>> [numero_dedos_comprobacion1 numero_dedos_comprobacion2 numero_dedos]

ans =

     4     3     4
```

Figura 5.10: Ejemplo 3 de contador de dedos.

5.2. Controlabilidad

La controlabilidad, o lo que es lo mismo, la capacidad de poder manejar el robot puede llegar a ser un tanto compleja. Esto se debe a que el tiempo de procesado de imágenes es un tanto variable, y si a esto le sumamos que se ha integrado el módulo en el programa principal para que se tenga que detectar al menos tres veces el mismo gesto para que se envíe la orden al Mindstorm esto crea un desfase variable. Se han realizado pruebas con el fin de obtener un rango de tiempos de varios procesos. Estas pruebas se han realizado con un ordenador portátil, siendo tan sólo orientativas ya que dependerán del procesador con el que se trabaje.

Proceso	**Rango tiempos estándar**	**Tiempo máximo**
Hacer foto	0.04 - 0.07	0.08
Tratamiento imagen	0.05 - 0.08	0.11
Comando a motor	0.015 - 0.03	0.04
Bucle (gesto de freno)	0.043 - 0.24	0.41
Bucle (otros gestos)	0.16 - 0.45	0.79

Cuadro 5.1: Resumen de tiempo de respuesta de acciones y procesos.

Cabe destacar que las dos últimas filas de la tabla lo que denomina bucle es cada iteración que se realiza en el programa principal, lo que abarca todo el proceso de análisis de imagen y toma de desiciones. Se ha desglosado en dos partes, ya que en caso de que el programa detecte el gesto de freno no se ejecutará el programa contador de dedos, lo que reduce sensiblemente el tiempo de ejecución.

A los datos de la tabla hay que sumarle el desfase que provoca enviar el comando vía bluetooth y el procesado del mismo por parte del Mindstorm. Debido a que se ha de repetir tres veces el mismo gesto de manera consecutiva para que se acepte ese gesto como válido se generará un desface que experimentalmente suele rondar el segundo. Este desfase con un poco de habilidad se puede corregir cambiando el gesto antes de estar el robot en la posición deseada, compensándose el desfase por la acción del usuario. Hay que destacar que debido a la velocidad del Mindstorm esta corrección no será muy difícil de realizar, pudiendo aprender a controlar los movimientos en unos pocos intentos. Otro problema de este desfase es que hace que haya un tiempo mínimo de tres repeticiones en el análisis de las imágenes, lo cual supone que si se ejecuta la señal haya un tiempo mínimo de ejecución del comando, es decir, habrá un giro mínimo que se pueda realizar, al igual que una distancia lineal mínima. Experimentalmente el giro mínimo se puede comprender entre 10 y 15 grados y la distancia lineal a velocidad normal es de 5 a 10 centímetros.

5.3. Movilidad

A continuación se muestran una serie de imágenes en las que se puede apreciar el movimiento del robot Mindstorm NXT mediante una secuencia

de fotogramas. En el primer caso se trata del movimiento de avance y en el segundo se puede apreciar el giro hacia la derecha, es decir, el robot gira a favor de las agujas del reloj.

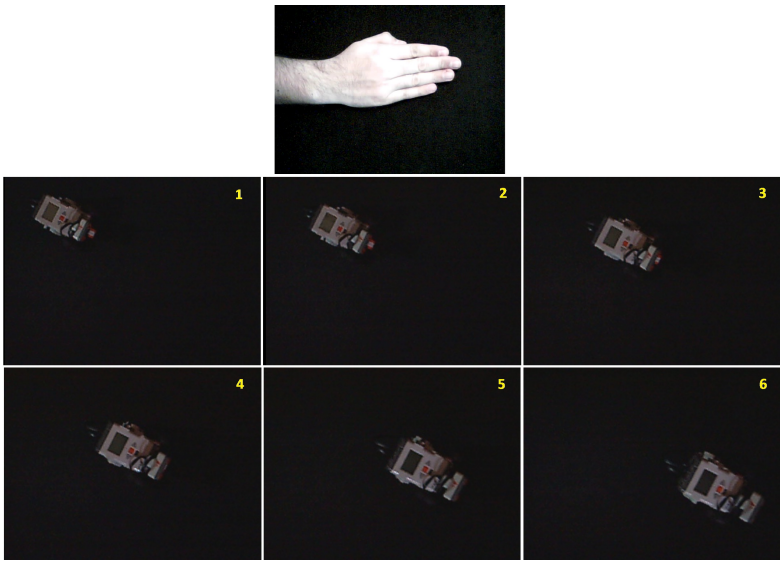

Figura 5.11: Ejemplo de avance del robot.

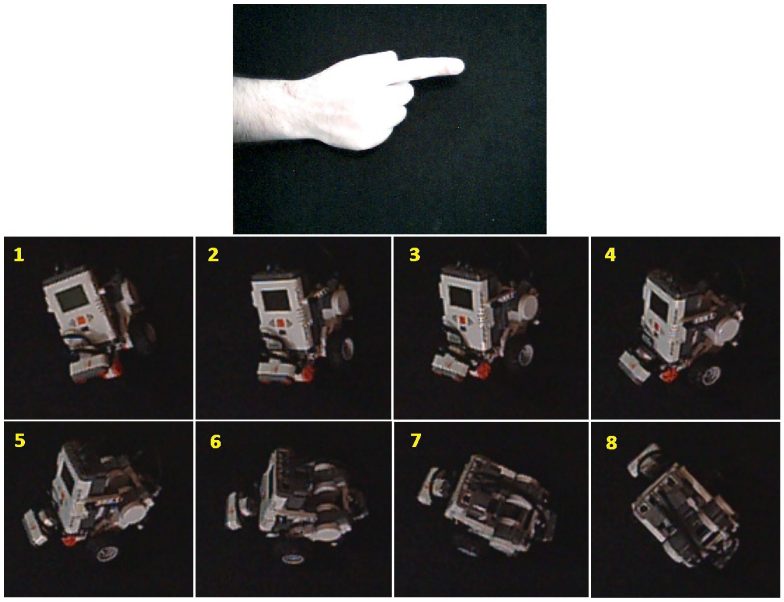

Figura 5.12: Ejemplo de rotación del robot hacia la derecha.

A continuación se ilustra un movimiento más complejo, creándose un movimiento fluido. Con el fin de no ir poniendo parejas de imágenes tan sólo se ha mostrado una vez la imagen del gesto y en la parte derecha se ha puesto el movimiento paso a paso del vehículo.

Figura 5.13: Ejemplo de un movimiento continuo.

CAPÍTULO 6

CONCLUSIONES

Este capítulo se crea con el fin de mostrar las conclusiones obtenidas para el presente proyecto de final de carrera. Éstas se dividirán en tres, evaluando por separado cada parte del proyecto dependiendo de su ámbito, de manera que se pueda profundizar de manera apreciable en todos los puntos del diseño y realización del mismo.

6.1. Estructurales

Desde el punto de vista estructural, y teniendo en cuenta que el diseño se ha realizado para interiores, el modelo seleccionado cumple su misión perfectamente, facilitando el control a la hora de realizar los movimientos. Debido a su tamaño se tiene la ventaja de una gran maniobrabilidad pudiendo hacer recorridos en zonas con numerosos obstáculos, sin embargo, este reducido tamaño hace que no sea viable su manejo en superficies no uniformes o abruptas, como podría ser una zona de tierra, ya que las piedras o simplemente el polvo podrían hacer que la rueda loca se bloquee dificultando el movimiento y perdiendo movilidad.

Otra conclusión que se puede obtener es la utilidad del sensor de ultrasonido con el cual se pueden evitar choques en caso de pérdida de visión del Mindstorm NXT o de una mala elección de gesto por parte del usuario durante el manejo. Viendo la utilidad de este sensor, pese a su medición bastante errática, se podría considerar la inclusión de algún otro sensor con la finalidad de aumentar la funcionalidad del robot como, por ejemplo, el micrófono con el cual se podría poner una parada de emergencia detectada por sonido.

6.2. Visión artificial por ordenador

En cuanto a la visión artificial, para la cual se ha realizado el programa explicado en el cuarto tema de la presente memoria, se ha comprobado experimentalmente que cumple el objetivo que se tenía asignado, que era la teleoperación mediante gestos de la mano. En cuanto al programa realizado se ha probado con fondos lisos, tanto claro (blanco o gris) como oscuro (negro), donde cabe destacar que en el caso de usar un fondo claro es necesaria una buena iluminación ya que si la luz es oblicua a la mano se producen sombras que serán reconocidas como si pertenecieran al gesto produciéndose posiblemente errores en el reconocimiento de los patrones. También se han probado fondos no lisos en los que los colores difieran claramente del color de la mano, ya que si hay colores similares en el fondo, como puede ser el color marrón de la madera, se producen errores en el tratamiento de la imagen de manera que es deseable evitar que el fondos tenga estos colores. Para los dos casos de fondos teniendo en cuenta las indicaciones mencionadas se consigue reconocer los gestos, y por consiguiente, el control del Mindstorm NXT.

6.3. Funcionamiento: manejo y controlabilidad

En cuanto al funcionamiento, en tiempo real, del Mindstorm NXT se puede decir que hace falta práctica para poder controlarlo de la manera deseada ya que el retardo entre el gesto, su reconocimiento y el envío del comando pertinente por bluetooth provoca un desfase en la reacción del robot que hay que tener en cuenta a la hora de ir cambiando el gesto de la mano.

Una vez se haya practicado lo suficiente se consigue un buen control del robot pudiendo realizar circuitos de relativa complicación. Por otro lado la maniobrabilidad debido a su tamaño y a la capacidad de poder girar sobre si mismo hacen que el Mindstorm pueda realizar complicados trayectos sin la necesidad de tener que recibir ayuda externa.

Tras el análisis realizado en los distintos apartados se puede concluir que se ha conseguido el objetivo buscado en este proyecto final de carrera, la teleoperación del Mindstorm mediante técnicas de visión artificial.

CAPÍTULO 7

TRABAJOS FUTUROS

Debido a la flexibilidad en el diseño del Mindstorm NXT se pueden hacer numerosos proyectos futuros tanto variantes de este como realizar de manera similar este proyecto. A continuación se muestran posibles puntos a mejorar del actual proyecto.

Pese a que para este proyecto no se ha considerado necesario realizar una red neuronal para lograr el objetivo establecido un posible proyecto futuro es su integración en el proyecto, de manera que se pudieran realizar más gestos de la mano pudiendo aumentar su complejidad.

Otro posible trabajo futuro es la adición de un gesto con el cual el Mindstorm pueda navegar de manera independiente mediante el uso de varios sensores de ultrasonido, pudiendo ir creando un mapa del entorno por el que se mueve basado en las mediciones de los sensores.

Adaptar el sistema para poder utilizarlo de manera portable, es decir, no tener que depender de un ordenador y una cámara web para realizar la teleoperación. Consistiría en poder realizar la visión artificial en tiempo real mediante las imágenes obtenidas mediante la cámara de un teléfono móvil

pudiendo manejar el robot de manera remota con el bluetooth del mismo teléfono.

Otra posibilidad alternativa de la teleoperación es variar la forma de visión artificial, pudiendo realizarla mediante otras técnicas o usando otros complementes como puede ser el Kinect de la Xbox360 con el cual se puede detectar los movimientos de un individuo de forma que se pueda mover el robot o con el movimiento únicamente de la mano o con movimientos de todo el cuerpo.

BIBLIOGRAFÍA

[1] Página web oficial de Mindstorm de la Universidad de Aachen. «*http://www.mindstorms.rwth-aachen.de/*».

[2] Página web oficial de la toolbox del Mindstorm de Matlab. «*http://www.mathworks.es/academia/lego-mindstorms-nxt-software/legomindstorms-matlab.html*».

[3] Página web oficial de Lego. «*http://mindstorms.lego.com/en-us/bluetooth/Default.aspx?domainredir=www.mindstorms.com*».

[4] Página web de la toolbox de adquisición de imágenes. «*http://www.mathworks.com/help/toolbox/imaq/*».

[5] Vélez Serrano, J.F. «*Visión por computador*».

[6] Gómez de Gabriel, J. M. «*Teleoperación y telerrobótica*».

[7] Pinto Bermúdez, E. «*Fundamentos de control con MATLAB*».

[8] Bunke, H. «*Progress in computer vision and image analysis*».

[9] Tou, J.T. y González, R.C. «*Pattern Recognition Principles*».